4-4-2003

On Learning of Functions Refutably

Sanjay Jain
National University of Singapore

Efim Kinber
Sacred Heart University, kinbere@sacredheart.edu

Rolf Wiehagen

Thomas Zeugmann

On Learning of Functions Refutably

Sanjay Jain [a,1] Efim Kinber [b] Rolf Wiehagen [c]
Thomas Zeugmann [d]

[a] *School of Computing, National University of Singapore, Singapore 119260, Singapore, Email:sanjay@comp.nus.edu.sg*

[b] *Department of Computer Science, Sacred Heart University, Fairfield, CT 06432-1000, U.S.A., Email: kinbere@sacredheart.edu*

[c] *Department of Computer Science, University of Kaiserslautern, D-67653 Kaiserslautern, Germany, Email: wiehagen@informatik.uni-kl.de*

[d] *Medizinische Universität Lübeck, Institut für Theoretische Informatik, Wallstraße 40, 23560 Lübeck, Germany, Email: thomas@tcs.mu-luebeck.de*

Abstract

Learning of recursive functions refutably informally means that for *every* recursive function, the learning machine has either to learn this function or to refute it, that is to signal that it is not able to learn it. Three modi of making precise the notion of refuting are considered. We show that the corresponding types of learning refutably are of strictly increasing power, where already the most stringent of them turns out to be of remarkable topological and algorithmical richness. Furthermore, all these types are closed under union, though in different strengths. Also, these types are shown to be different with respect to their intrinsic complexity; two of them do not contain function classes that are "most difficult" to learn, while the third one does. Moreover, we present several characterizations for these types of learning refutably. Some of these characterizations make clear where the refuting ability of the corresponding learning machines comes from and how it can be realized, in general.

For learning with anomalies refutably, we show that several results from standard learning without refutation stand refutably. From this we derive some hierarchies for refutable learning. Finally, we prove that in general one cannot trade stricter refutability constraints for more liberal learning criteria.

[1] Supported in part by NUS grant number RP3992710.

1 Introduction

The basic scenario in learning theory informally consists in that a learning machine has to learn some unknown object based on certain information, that is the machine creates one or more hypotheses which eventually converge to a more or less correct and complete description of the object. In learning *refutably* the main goal is more involved. Here, for *every* object from a given universe, the learning machine has either to learn the object or to refute it, that is to "signal" if it is incapable to learn this object. This approach is philosophically motivated by Popper's logic of scientific discovery (testability, falsifiability, refutability of scientific hypotheses), see [33] and [25], for a more detailed discussion. Moreover, this approach has also some rather practical implications. Indeed, if the learning machine informs a user of its inability to learn a certain object, then the user can react upon this inability, by modifying the machine, by changing the space of hypotheses, or by weakening the learning requirements, for example.

A crucial point of learning refutably is to formally define how the machine is allowed or required to refute a non-learnable object. In the ground-breaking paper by Mukouchi and Arikawa [30] required that refuting takes place in a "one shot" manner, that is, if after some finite amount of time, the machine comes to the conclusion that it is not able to learn the object under consideration, then it outputs a special "refuting symbol" and stops the learning process forever. Two weaker possibilities of refuting are based on the following observation. Suppose that at some time, the machine feels unable to learn the unknown object and signals this by outputting the refuting symbol. Nevertheless, this time the machine keeps trying to learn this object. It may happen that the information it further receives contains a new evidence which leads to changing its mind about its inability to learn the object. Of course, this process of "alternations" can repeat. And it may end in learning the object. Or it may end in refuting it by never revising the machine's belief in its inability to learn the object, or, equivalently, by forever outputting the refuting symbol from some point on. Or, finally, there may be infinitely many such alternations between trying to learn and believing that this is impossible. In our paper, we will allow and study all three of these modes of learning refutably.

Our universe is the class $\mathcal{R}$ of all recursive functions, i.e., all computable functions being defined everywhere. The basic learning criterion used will be **Ex**, learning in the limit, see Definition 1. We then consider the following types of learning refutably:

RefEx, where refuting a non-learnable function takes place in the *one shot* manner described above (cf. Definition 5).

WRefEx, where both learning and refuting are *limiting* processes, that is on every function from the universe, the learning machine converges either to a correct hypothesis for this function or to the refuting symbol, see Definition 6 (**W** stands for "weak").

RelEx, where a function is considered as being refuted if the learning machine outputs the refuting symbol *infinitely often* on this function, see Definition 7, (**Rel** stands for "reliable", since this type coincides with so-called reliable learning, as we shall see below).

As it immediately follows from the definitions of all these types of learning refutably, every function from our universe will indeed either be learned or be refuted by every machine that learns refutably. In other words, it can *not* happen that such a machine converges to an *incorrect* hypothesis, see Correctness Lemma below. Thus, this lemma can be viewed as a justification for the above approaches of refutable learning.

We then show that these types of learning refutably are of strictly increasing power, see Theorem 16. Already the most stringent of them, **RefEx**, turns out to be of remarkable topological and algorithmical richness (cf. Proposition 11 and Corollary 30). Furthermore, all of these learning types are closed under union, Proposition 18, where **RefEx** and **WRefEx**, on the one hand, and **RelEx**, on the other hand, do not behave completely analogous. Such a difference can also be exhibited with respect to the so-called intrinsic complexity; actually, both **RefEx** and **WRefEx** do not contain function classes that are "most difficult" to learn, while **RelEx** does contain such classes, see Theorems 26 and 27, respectively. Moreover, we present several characterizations for our types of learning refutably. Specifically, some of these characterizations make it clear where the refuting ability of the corresponding learning machines comes from and how it can be realized, in general (cf. Theorems 39 and 44).

Besides pure **Ex**-learning refutably we also consider **Ex**-learning with anomalies as well as **Bc**-learning with anomalies refutably (cf. Definitions 49 and 50). We show that many results from standard learning without refutation stand refutably. From this we derive several hierarchies for refutable learning, thereby solving an open problem from [23], see Corollaries 56 and 67. Moreover, we prove that, in general, one cannot trade a stricter refutability constraint for a more liberal learning criterion (cf. Corollary 71 and Theorem 72).

Since the pioneering paper [30] learning with refutation has attracted much attention. The line initiated by [30], i.e., studying learning with refutation for indexed families of *languages*, was also applied to learning of elementary formal systems in [31]. As a consequence of this model, if an indexed family of recursive languages can be refutably learned from text then this class cannot contain any infinite language. This limitation led Lange and Watson [25] to

consider a more tolerant approach. In their model a refuting learning machine is no longer required to refute *every* text describing a language outside the class to be learned. Instead, the machine has to refute only such texts containing a finite sample being not contained in any language from the class to be learned. This indeed leads to a richer spectrum of indexed families of recursive languages that are learnable with so-called justified refutation. Jain [18] then generalized the study in two directions. First, classes of arbitrary recursively enumerable languages were considered. Second, the learning machine was allowed either to refute or to learn unrepresentative texts. For a natural interpretation of "unrepresentative", the power of the justified refutation model has been shown to reach the power of the unrestricted model of learning languages from text.

Learning *functions* with refutation was considered by Miyahara [29] for indexed classes of primitive recursive functions. For arbitrary classes of arbitrary recursive functions, an alternative approach has been developed and studied by Jantke [20] and Grieser [17]. In a sense, their approach is orthogonal to ours. Actually, on the one hand, their model allows to learn richer classes than we can. On the other hand, in certain cases every machine that learns such a richer class converges to incorrect hypotheses on infinitely many functions outside this class, whereas in our approach the Correctness Lemma guarantees that no machine converges incorrectly on whatever function from the universe.

The paper is organized as follows. Section 2 provides the necessary notations and definitions, as well as the Correctness Lemma. Section 3 deals with **Ex**-learning refutably. In Section 4, we consider **Ex**- and **Bc**-learning with anomalies refutably.

2 Notation and Preliminaries

Recursion-theoretic concepts not explained below are treated in [35]. $\mathbb{N}$ denotes the set of natural numbers. Furthermore, $i \dot{-} j$ is defined as follows:

$$i \dot{-} j = \begin{cases} i - j, & \text{if } i \geq j; \\ 0, & \text{otherwise.} \end{cases}$$

Let $\in$, $\subseteq$, $\subset$, $\supseteq$, $\supset$, respectively, denote the membership, subset, proper subset, superset and proper superset relations for sets. The empty set is denoted by $\emptyset$. We let $\mathrm{card}(S)$ denote the cardinality of the set S. The minimum and maximum of a set S are denoted by $\min(S)$ and $\max(S)$, respectively. We take $\max(\emptyset)$ to be 0 and $\min(\emptyset)$ to be ∞.

$\langle \cdot, \cdot \rangle$ denotes a 1-1 computable mapping from pairs of natural numbers onto $\mathbb{N}$. π_1, π_2 are the corresponding projection functions. $\langle \cdot, \cdot \rangle$ is extended to n-tuples of natural numbers in a natural way. η, with or without subscripts, superscripts, primes and the like, ranges over partial functions. If η_1 and η_2 are both undefined on input x, then, we take $\eta_1(x) = \eta_2(x)$. We say that $\eta_1 \subseteq \eta_2$ iff for all x in the domain of η_1, $\eta_1(x) = \eta_2(x)$. We let domain(η) and range(η), respectively, denote the domain and range of the partial function η. $\eta(x)\downarrow$ and $\eta(x) = \downarrow$ both denote that $\eta(x)$ is defined and $\eta(x)\uparrow$ as well as $\eta(x) = \uparrow$ stand for $\eta(x)$ is undefined. We identify a partial function η with its graph $\{(x, \eta(x)) \mid x \in \text{domain}(\eta)\}$.

For $r \in \mathbb{N}$, the r-extension of η denotes the function f defined as follows:

$$f(x) = \begin{cases} \eta(x), & \text{if } x \in \text{domain}(\eta); \\ r, & \text{otherwise.} \end{cases}$$

We write r-ext(η) for the r-extension of η. $\mathcal{R}$ denotes the class of all *recursive* functions, i.e., total computable functions with arguments and values from $\mathbb{N}$. By $\mathcal{R}_{0,1}$ we denote the class of all recursive functions with range contained in $\{0, 1\}$. $\mathcal{C}$ and $\mathcal{S}$, with or without subscripts, superscripts, primes and the like, range over subsets of $\mathcal{R}$. For $\mathcal{C} \subseteq \mathcal{R}$, we let $\overline{\mathcal{C}}$ denote $\mathcal{R} \setminus \mathcal{C}$. $\mathcal{P}$ denotes the class of all *partial recursive* functions over $\mathbb{N}$. f, g, h and F, with or without subscripts, superscripts, primes and the like, range over recursive functions unless otherwise specified.

A computable numbering (or just numbering) is a partial recursive function of two arguments. For a numbering $\psi(\cdot, \cdot)$, we use ψ_i to denote the function $\lambda x.\psi(i, x)$. In other words, ψ_i is the function computed by the program i in the numbering ψ. ψ and ϱ range over numberings. $\mathcal{P}_\psi$ denotes the set of partial recursive functions in the numbering ψ, i.e., $\mathcal{P}_\psi = \{\psi_i \mid i \in \mathbb{N}\}$. We set $\mathcal{R}_\psi = \{\psi_i \mid i \in \mathbb{N} \ \& \ \psi_i \in \mathcal{R}\}$. That is, $\mathcal{R}_\psi$ stands for the set of all recursive functions in the numbering ψ. A numbering ψ is called one-to-one iff $\psi_i \neq \psi_j$ for any distinct i, j. Hence, for any $\eta \in \mathcal{P}_\psi$, there is exactly one ψ-index i such that $\psi_i = \eta$. φ denotes a *fixed* acceptable programming system (cf. [35]). We write φ_i for the partial recursive function computed by program i in the φ-system. We let Φ be an arbitrary Blum [6] complexity measure associated with the acceptable programming system φ; many such measures exist for any acceptable programming system. We assume without loss of generality that $\Phi_i(x) \geq x$, for all i, x. $\varphi_{i,s}$ is defined as follows:

$$\varphi_{i,s}(x) = \begin{cases} \varphi_i(x), & \text{if } x < s \text{ and } \Phi_i(x) < s; \\ \uparrow, & \text{otherwise.} \end{cases}$$

A class $\mathcal{C} \subseteq \mathcal{R}$ is said to be recursively enumerable iff there exists an r.e. set X such that $\mathcal{C} = \{\varphi_i \mid i \in X\}$. For any non-empty recursively enumerable

class $\mathcal{C}$, there exists a recursive function f such that $\mathcal{C} = \{\varphi_{f(i)} \mid i \in \mathbb{N}\}$.

A function g is said to be an *accumulation point* of a class $\mathcal{C} \subseteq \mathcal{R}$ iff $g \in \mathcal{R}$ and $(\forall n \in \mathbb{N})(\exists f \in \mathcal{C})[(\forall x \leq n)[g(x) = f(x)] \ \& \ f \neq g]$. Note that the accumulation point may or may not belong to the class. For $\mathcal{C} \subseteq \mathcal{R}$, we let $\mathrm{Acc}(\mathcal{C}) = \{g \mid g \text{ is an accumulation point of } \mathcal{C}\}$.

The quantifier $\forall^\infty$ denotes for all but finitely many; that is, $(\forall^\infty x)[P(x)]$ means $\mathrm{card}(\{x \mid \neg P(x)\}) < \infty$. The following functions and classes are commonly considered below. Zero is the everywhere 0 function, i.e., $\mathrm{Zero}(x) = 0$, for all $x \in \mathbb{N}$. $FINSUP = \{f \mid (\forall^\infty x)[f(x) = 0]\}$ denotes the class of all recursive functions of finite support.

2.1 Function Identification

We first describe inductive inference machines. We assume, that the graph of a function is fed to a machine in canonical order.

For a partial function η such that $\eta(x)$ is defined for all $x < n$, we write $\eta[n]$ for the set $\{(x, \eta(x)) \mid x < n\}$, the finite initial segment of η of length n. Clearly, $\eta[0]$ denotes the empty segment. SEG denotes the set of all finite initial segments, i.e., $\{f[n] \mid f \in \mathcal{R} \ \& \ n \in \mathbb{N}\}$. Furthermore, we set $\mathrm{SEG}_{0,1} = \{f[n] \mid f \in \mathcal{R}_{0,1} \ \& \ n \in \mathbb{N}\}$. We let σ, τ and γ, with or without subscripts, superscripts, primes and the like, range over SEG. Λ denotes the empty segment. We assume some computable ordering of elements of SEG. Thus, one can talk about recursively enumerable subsets of SEG and comparison among members of SEG, that is $\sigma < \tau$, if σ appears before τ in this ordering. Similary one can talk about the least element of a subset of SEG.

Let $|\sigma|$ denote the length of σ. Thus, $|f[n]| = n$, for every total function f and all $n \in \mathbb{N}$. If $|\sigma| \geq n$, then we let $\sigma[n]$ denote $\{(x, \sigma(x)) \mid x < n\}$. An *inductive inference machine* (IIM) is an algorithmic device that computes a total mapping from SEG into $\mathbb{N}$ (cf. [15]). Since the set of all finite initial segments, SEG, can be coded onto $\mathbb{N}$, we can view these machines as taking natural numbers as input and emitting natural numbers as output. We say that $\mathbf{M}(f)$ converges to i (written: $\mathbf{M}(f)\downarrow = i$) iff $(\forall^\infty n)[\mathbf{M}(f[n]) = i]$; $\mathbf{M}(f)$ is undefined if no such i exists. The next definitions describe several criteria of function identification.

Definition 1 (Gold [15]). *Let $f \in \mathcal{R}$ and let $\mathbf{M}$ be an IIM.*

(a) $\mathbf{M}$ **Ex**-identifies f *(written: $f \in \mathbf{Ex}(\mathbf{M})$) just in case there exists an $i \in \mathbb{N}$ such that $\mathbf{M}(f)\downarrow = i$ and $\varphi_i = f$.*
(b) $\mathbf{M}$ **Ex**-identifies $\mathcal{C}$ *iff* $\mathbf{M}$ **Ex**-*identifies each $f \in \mathcal{C}$.*
(c) $\mathbf{Ex} = \{\mathcal{C} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{C} \subseteq \mathbf{Ex}(\mathbf{M})]\}$.

By the definition of convergence, only finitely many data points from a function f have been observed by an IIM $\mathbf{M}$ at the (unknown) point of convergence. Hence, some form of learning must take place in order for $\mathbf{M}$ to learn f. For this reason, hereafter the terms *identify*, *learn* and *infer* are used interchangeably.

Definition 2 (Bārzdiņš [2], Case and Smith [10]). *Let $f \in \mathcal{R}$ and let $\mathbf{M}$ be an IIM.*

(a) $\mathbf{M}$ **Bc**-identifies f *(written: $f \in \mathbf{Bc}(\mathbf{M})$) iff, for all but finitely many $n \in \mathbb{N}$, $\mathbf{M}(f[n])$ is a program for f, i.e., $\varphi_{\mathbf{M}(f[n])} = f$.*
(b) $\mathbf{M}$ **Bc**-identifies $\mathcal{C}$ *iff* $\mathbf{M}$ **Bc**-*identifies each $f \in \mathcal{C}$.*
(c) $\mathbf{Bc} = \{\mathcal{C} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{C} \subseteq \mathbf{Bc}(\mathbf{M})]\}$.

Definition 3 (Minicozzi [28], Blum and Blum [5]). *Let $\mathbf{M}$ be an IIM.*

(a) $\mathbf{M}$ *is* reliable *iff for all $f \in \mathcal{R}$, $\mathbf{M}(f)\downarrow \Rightarrow$ $\mathbf{M}$* **Ex**-*identifies f.*
(b) $\mathbf{M}$ **RelEx**-identifies $\mathcal{C}$ *(written: $\mathcal{C} \subseteq \mathbf{RelEx}(\mathbf{M})$) iff $\mathbf{M}$ is reliable and $\mathbf{M}$* **Ex**-*identifies $\mathcal{C}$.*
(c) $\mathbf{RelEx} = \{\mathcal{C} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathbf{M}$ **RelEx**-*identifies $\mathcal{C}$*$]\}$.

Thus, intuitively, a machine is reliable if it does not converge on functions it fails to identify. For further references on reliable learning besides [28,5], see [22,16,23,41,8].

Definition 4 $\mathbf{NUM} = \{\mathcal{C} \mid (\exists \mathcal{C}' \mid \mathcal{C} \subseteq \mathcal{C}' \subseteq \mathcal{R})[\mathcal{C}'$ *is recursively enumerable*$]\}$.

For references on inductive inference within **NUM**, the set of all recursively enumerable classes and their subclasses, the reader is referred to [15,3,12]. For references surveying the general theory of learning recursive functions, we refer the reader to [1,5,10,11,24,32,19].

2.2 Learning Refutably

In this subsection we introduce learning with refutation. The idea is that the learning machine should "refute" functions which it does not identify. We consider three versions of refutation based on how the machine is required to refute a function. First we need to extend the definition of IIM to allow a machine to output a special symbol $\perp$. Thus, now an IIM is a mapping from SEG to $\mathbb{N} \cup \{\perp\}$. Convergence of an IIM on a function can be defined as before (where a machine may now converge to a natural number or to $\perp$).

Definition 5 *Let $\mathbf{M}$ be an IIM.* $\mathbf{M}$ **RefEx**-identifies *a class $\mathcal{C}$ (written: $\mathcal{C} \subseteq \mathbf{RefEx}(\mathbf{M})$) iff the following conditions are satisfied.*

(a) $\mathcal{C} \subseteq \mathbf{Ex}(\mathbf{M})$.
(b) *For all $f \in \mathbf{Ex}(\mathbf{M})$, for all n, $\mathbf{M}(f[n]) \neq \perp$.*

(c) *For all* $f \in \mathcal{R}$ *such that* $f \notin \mathbf{Ex}(\mathbf{M})$*, there exists an* n *such that* $(\forall m < n)[\mathbf{M}(f[m]) \neq \perp]$ *and* $(\forall m \geq n)[\mathbf{M}(f[m]) = \perp]$.

Intuitively, for **RefEx**-identification, the IIM $\mathbf{M}$ outputs the special symbol $\perp$ on input function f to indicate that it is not going to **Ex**-identify f.

The following generalization of **RefEx** places less restrictive constraint on how the machine refutes a function. **WRef** below stands for weak refutation.

Definition 6 *Let* $\mathbf{M}$ *be an IIM.* $\mathbf{M}$ **WRefEx**-identifies *a class* $\mathcal{C}$ *(written:* $\mathcal{C} \subseteq \mathbf{WRefEx}(\mathbf{M})$*) iff the following conditions are satisfied.*

(a) $\mathcal{C} \subseteq \mathbf{Ex}(\mathbf{M})$.
(b) *For all* $f \in \mathcal{R}$ *such that* $f \notin \mathbf{Ex}(\mathbf{M})$, $\mathbf{M}(f)\downarrow = \perp$.

For weakly refuting a function, the machine just needs to converge to the refutation symbol $\perp$. Before convergence, it may change its mind finitely many times whether or not it is going to refute the function.

There is another possible way a machine may refute a function f, i.e., it outputs $\perp$ on f infinitely often. This version actually turns out to be equivalent to **RelEx**-learning considered above.

Definition 7 *Let* $\mathbf{M}$ *be an IIM.* $\mathbf{M}$ $\mathbf{RelEx}'$-identifies *a class* $\mathcal{C}$ *(written:* $\mathcal{C} \subseteq \mathbf{RelEx}'(\mathbf{M})$*) iff the following conditions are satisfied.*

(a) $\mathcal{C} \subseteq \mathbf{Ex}(\mathbf{M})$.
(b) *For all* $f \in \mathcal{R}$ *such that* $f \notin \mathbf{Ex}(\mathbf{M})$*, there exist infinitely many* n *such that* $\mathbf{M}(f[n]) = \perp$.

Proposition 8 $\mathbf{RelEx} = \mathbf{RelEx}'$.

Proof. We first show that $\mathbf{RelEx} \subseteq \mathbf{RelEx}'$.

Suppose $\mathbf{M}$ **RelEx**-identifies $\mathcal{C}$. Define $\mathbf{M}'$ as follows:

$$\mathbf{M}'(f[n]) = \begin{cases} \mathbf{M}(f[n]), & \text{if } n = 0 \text{ or } \mathbf{M}(f[n-1]) = \mathbf{M}(f[n]); \\ \perp, & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}'$ $\mathbf{RelEx}'$-identifies $\mathcal{C}$.

We now show that $\mathbf{RelEx}' \subseteq \mathbf{RelEx}$. Suppose $\mathbf{M}$ $\mathbf{RelEx}'$-identifies $\mathcal{C}$. Define $\mathbf{M}'$ as follows:

$$\mathbf{M}'(f[n]) = \begin{cases} \mathbf{M}(f[n]), & \text{if } \mathbf{M}(f[n]) \neq \perp; \\ n, & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}'$ **Ex**-identifies each f **Ex**-identified by $\mathbf{M}$. Also, if $\mathbf{M}$ outputs $\perp$ on f infinitely often, then $\mathbf{M}(f)\uparrow$, since it outputs arbitrarily

large numbers on f. It follows that $\mathbf{M}'$ **RelEx**-identifies $\mathcal{C}$. ∎

As it immediately follows from their definitions, for any of the learning types **RefEx**, **WRefEx** and **RelEx**, we get that any recursive function has either to be learned or to be refuted. We make this point formally precise by stating the following Correctness Lemma. Informally, this lemma says that for *every* type of learning refutably and for *every* machine that learns in the corresponding sense, one can trust in the correctness of *every* hypothesis from $\mathbb{N}$ the machine may converge to.

Lemma 9 (*Correctness Lemma*). *Let* $\mathbf{I} \in \{\mathbf{RefEx}, \mathbf{WRefEx}, \mathbf{RelEx}\}$. *For any* $\mathcal{C} \subseteq \mathcal{R}$, *any IIM* $\mathbf{M}$ *with* $\mathcal{C} \subseteq \mathbf{I}(\mathbf{M})$, *and any* $f \in \mathcal{R}$, *if* $\mathbf{M}(f)\downarrow \in \mathbb{N}$, *then* $\varphi_{\mathbf{M}(f)} = f$.

Proof. We prove the lemma here for **RefEx**. The remaining cases can be handled analogously. Let $\mathcal{C} \subseteq \mathcal{R}$ and let $\mathbf{M}$ be any IIM such that $\mathcal{C} \subseteq \mathbf{RefEx}$. Furthermore, let $f \in \mathcal{R}$ and assume that $\mathbf{M}(f)\downarrow \in \mathbb{N}$. Thus, we can conclude that $f \in \mathbf{Ex}(\mathbf{M})$. Otherwise condition (c) in Definition 5 must have happened, and hence $M(f[m]) = \perp$ for all but finitely many m, a contradiction to $\mathbf{M}(f)\downarrow \in \mathbb{N}$. Finally, by Definition 1, part (a), we directly obtain $\varphi_{\mathbf{M}(f)} = f$. ∎

Using essentially the idea from Gold [14], (for **Ex**-identification), for $\mathbf{I}$ being any of the learning criteria considered in this paper, one can show that:

> There exists an r.e. sequence $\mathbf{M}_0, \mathbf{M}_1, \mathbf{M}_2, \ldots$ of *total* inductive inference machines such that, for all $\mathcal{C} \in \mathbf{I}$, there exists an $i \in \mathbb{N}$ such that $\mathcal{C} \subseteq \mathbf{I}(\mathbf{M}_i)$.

In the following, we assume $\mathbf{M}_0, \mathbf{M}_1, \mathbf{M}_2, \ldots$ to be one such sequence of machines.

3 Ex-Learning Refutably

In this section, we first derive several properties of the types of learning refutably defined above. We then relate these types by their so-called intrinsic complexity. Finally, we present several characterizations for refutable learnability.

We start with exhibiting some properties of the classes being learnable refutably. Specifically, these properties imply that the corresponding learning types are of strictly increasing power, where already the most stringent of these types, **RefEx**, turns out to be of surprising richness. The first of these properties consists in that any class from **RefEx** can be enriched by including all of its accumulation points. Note that, in general, this is not possible for the classes from **WRefEx** and **RelEx**, as it immediately follows from the proof of Theorem 16.

Proposition 10 *For any* $\mathcal{C} \in$ **RefEx**, $\mathcal{C} \cup \mathrm{Acc}(\mathcal{C}) \in$ **RefEx**.

Proof. Informally, the result follows from the fact that in any type of refutable learning, any function $f \in \mathcal{R}$, will either be identified or refuted. Since in **RefEx**-learning an accumulation point can never be refuted, it has to be learned. Formally, suppose $\mathcal{C} \in$ **RefEx** as witnessed by total IIM **M**. Suppose $g \in \mathcal{R}$ is an accumulation point of $\mathcal{C}$. We claim that **M** must **Ex**-identify g. Assume to the contrary that for some n, $\mathbf{M}(g[n]) = \perp$. Then, by the definition of accumulation point, there is a function $f \in \mathcal{C}$ such that $g[n] \subseteq f$. Hence $\mathbf{M}(f[n]) = \perp$, too, a contradiction to **M** **RefEx**-identifying $\mathcal{C}$. ∎

The next proposition shows that **RefEx** contains "topologically rich", namely *non-discrete* classes, i.e. classes which contain accumulation points. Thus, **RefEx** is "richer" than usual **Ex**-learning without any mind change, since any class being learnable in that latter sense may not contain any of its accumulation points (cf. [26]). More precisely, **RefEx** and **Ex**-learning without mind changes are set-theoretically *incomparable*; the missing direction easily follows from Theorem 53 below.

Proposition 11 **RefEx** *contains non-discrete classes.*

Proof. For $i \in \mathbb{N}$, define f_i as follows.

$$f_i(x) = \begin{cases} 0, & \text{if } x < i; \\ 1, & \text{otherwise.} \end{cases}$$

Let $\mathcal{C} = \{f_i \mid i \in \mathbb{N}\} \cup \{\mathrm{Zero}\}$. Then, clearly, $\mathcal{C}$ is non-discrete and **RefEx**-learnable. ∎

The following proposition establishes some bound on the topological richness of the classes from **WRefEx**.

Definition 12 *A class* $\mathcal{C} \subseteq \mathcal{R}$ *is called* initially complete *iff for every* $\sigma \in$

SEG, *there is a function* $f \in \mathcal{C}$ *such that* $\sigma \subseteq f$.

Proposition 13 **WRefEx** *does not contain any initially complete class.*

Proof. Assume to the contrary that there is an initially complete class $\mathcal{C}$ that is **WRefEx**-learnable by some total IIM $\mathbf{M}$.

Claim 14 *For all* $\sigma \in$ SEG, *there exists a* $\tau \in$ SEG *such that* $\sigma \subseteq \tau$, *and* $\mathbf{M}(\sigma) \neq \mathbf{M}(\tau)$.

Proof. If for all extensions τ of σ, $\mathbf{M}(\sigma) = \mathbf{M}(\tau)$, then $\mathbf{M}$ can **Ex**-identify at most one extension of σ. But then $\mathcal{C}$ is not initially complete. □

Now, let σ_i, $i \in \mathbb{N}$, be defined such that σ_i can be obtained effectively from i, $\sigma_0 = \Lambda$, and for all i, $\sigma_i \subseteq \sigma_{i+1}$ and $\mathbf{M}(\sigma_i) \neq \mathbf{M}(\sigma_{i+1})$. Note that this is possible due to Claim 14. Now let $f = \bigcup_{i \in \mathbb{N}} \sigma_i$. Clearly, $f \in \mathcal{R}$, but $\mathbf{M}$ on f makes infinitely many mind changes. Thus, $\mathbf{M}$ neither **Ex**-identifies, nor refutes f. Thus, $\mathbf{M}$ does not **WRef**-identify $\mathcal{C}$. ∎

The following result is needed for proving Theorem 16 below.

Lemma 15 $\mathcal{C} = \{f \in \mathcal{R} \mid (\forall x \in \mathbb{N})[f(x) \neq 0]\} \notin \mathbf{Ex}$.

Proof. Suppose by way of contradiction $\mathbf{M}$ **Ex**-identifies $\mathcal{C}$. For any $\sigma \in$ SEG, let τ_σ be defined as follows:

$$\tau_\sigma(x) = \begin{cases} \sigma(x) + 1, & \text{if } \sigma(x)\downarrow; \\ \uparrow, & \text{otherwise.} \end{cases}$$

Let prog be a recursive function such that, for any program p, $\varphi_{\text{prog}(p)}(x) = \varphi_p(x) \dot{-} 1$. Note that by the s-m-n theorem, there exists such a recursive function prog. Now define $\mathbf{M}'(\sigma) = \text{prog}(\mathbf{M}(\tau_\sigma))$.

It is easy to verify that if $\mathbf{M}$ **Ex**-identifies $\mathcal{C}$, then $\mathbf{M}'$ **Ex**-identifies $\mathcal{R}$. However, $\mathcal{R} \notin \mathbf{Ex}$, see [15]. Thus $\mathcal{C} \notin \mathbf{Ex}$. ∎

We are now ready to prove that **RefEx**, **WRefEx** and **RelEx**, respectively, are of strictly increasing power.

Theorem 16 $\mathbf{RefEx} \subset \mathbf{WRefEx} \subset \mathbf{RelEx}$.

Proof. $\mathbf{RefEx} \subseteq \mathbf{WRefEx} \subseteq \mathbf{RelEx}$ follows easily from the definitions and Proposition 8.

We first show that $\mathbf{WRefEx} \setminus \mathbf{RefEx} \neq \emptyset$. For that purpose, we define $\mathrm{SEG}^+ = \{f[n] \mid f \in \mathcal{R} \ \& \ n \in \mathbb{N} \ \& \ (\forall x \in \mathbb{N})[f(x) \neq 0]\}$. Let $\mathcal{C} = \{0\text{-ext}(\sigma) \mid \sigma \in \mathrm{SEG}^+\}$. Then $\mathrm{Acc}(\mathcal{C}) = \{f \in \mathcal{R} \mid (\forall x \in \mathbb{N})[f(x) \neq 0]\}$, which is not in **Ex**, by Lemma 15. Thus, $\mathcal{C} \cup \mathrm{Acc}(\mathcal{C}) \notin \mathbf{Ex}$, and hence, $\mathcal{C} \notin \mathbf{RefEx}$, by Proposition 10.

In order to show that $\mathcal{C} \in \mathbf{WRefEx}$, let $\mathrm{prog} \in \mathcal{R}$ be a recursive function such that for any $\sigma \in \mathrm{SEG}^+$, $\mathrm{prog}(\sigma)$ is a φ-program for $0\text{-ext}(\sigma)$. Let $\mathbf{M}$ be defined as follows.

$$\mathbf{M}(f[n]) = \begin{cases} \perp, & \text{if } f[n] \in \mathrm{SEG}^+; \\ \mathrm{prog}(\sigma), & \text{if } 0\text{-ext}(f[n]) = 0\text{-ext}(\sigma), \text{ for some } \sigma \in \mathrm{SEG}^+; \\ \perp, & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}$ **WRefEx**-identifies $\mathcal{C}$.

We now show that $\mathbf{RelEx} \setminus \mathbf{WRefEx} \neq \emptyset$. Clearly, *FINSUP* is initially complete and $\mathit{FINSUP} \in \mathbf{NUM}$. Since $\mathbf{NUM} \subseteq \mathbf{RelEx}$, see [28], we have that $\mathit{FINSUP} \in \mathbf{RelEx}$. On the other hand, $\mathit{FINSUP} \notin \mathbf{WRefEx}$ by Proposition 13. ∎

As a consequence from the proof of Theorem 16, we can derive that the types **RefEx**, **WRefEx** and **RelEx** already differ on *recursively enumerable* classes.

Corollary 17 $\mathbf{RefEx} \cap \mathbf{NUM} \subset \mathbf{WRefEx} \cap \mathbf{NUM} \subset \mathbf{RelEx} \cap \mathbf{NUM}$.

Proof. Immediately from the proof of Theorem 16. ∎

We next point out that all the types of learning refutably share a pretty rare, but desirable property, namely to be closed under union.

Proposition 18 $\mathbf{RefEx}, \mathbf{WRefEx}$ *and* $\mathbf{RelEx}$ *are closed under union.*

Proof. In [28] it was shown that **RelEx** is closed under union. Now, suppose $\mathbf{I} \in \{\mathbf{RefEx}, \mathbf{WRefEx}\}$, $\mathcal{C} \subseteq \mathbf{I}(\mathbf{M}')$ and $\mathcal{S} \subseteq \mathbf{I}(\mathbf{M}'')$. Then, define an IIM $\mathbf{M}$ as follows.

$$\mathbf{M}(f[n]) = \begin{cases} \mathbf{M}'(f[n]), & \text{if } \mathbf{M}'(f[n]) \neq \perp; \\ \mathbf{M}''(f[n]), & \text{otherwise.} \end{cases}$$

Thus, informally, $\mathbf{M}$ simulates the first machine that currently does not refute the given function. It is easy to verify that $\mathcal{C} \cup \mathcal{S} \subseteq \mathbf{I}(\mathbf{M})$. ∎

Proposition 18 obviously applies also to the union of any finite number of classes. **RelEx** is even closed under the union of any effectively given *infinite* sequence of classes, see [28]. However, the latter is not true for both **RefEx** and **WRefEx**, as it can be seen by shattering the class *FINSUP* into its subclasses of *one* element each.

3.2 Intrinsic Complexity

There is another field where **RefEx** and **WRefEx**, on the one hand, and **RelEx**, on the other hand, behave differently, namely that of intrinsic complexity. The intrinsic complexity compares the difficulty of learning by using some reducibility notion, see [13]. As usual, with every reducibility notion comes a notion of completeness. Intuitively, a function class is complete for some learning type, if this class is "most difficult" to learn among all the classes from this learning type. As we will show, the types **RefEx** and **WRefEx** do not contain such complete classes, while **RelEx** does. We now proceed more formally.

Definition 19 *A sequence $P = p_0, p_1, \ldots$ of natural numbers is called* **Ex**-admissible *for $f \in \mathcal{R}$ iff P converges to a program p for f.*

Definition 20 (Rogers [35]). *A* recursive operator *is an effective total mapping, Θ, from (possibly partial) functions to (possibly partial) functions, which satisfies the following properties:*

(a) *Monotonicity: For all functions η, η', if $\eta \subseteq \eta'$ then $\Theta(\eta) \subseteq \Theta(\eta')$.*
(b) *Compactness: For all η, if $(x, y) \in \Theta(\eta)$, then there exists a finite function $\alpha \subseteq \eta$ such that $(x, y) \in \Theta(\alpha)$.*
(c) *Recursiveness: For all finite functions α, one can effectively enumerate (in α) all $(x, y) \in \Theta(\alpha)$.*

For each recursive operator Θ, we can effectively (from Θ) find a recursive operator Θ' such that

(d) for each finite function α, $\Theta'(\alpha)$ is finite, and its canonical index can be effectively determined from α, and
(e) for all total functions f, $\Theta'(f) = \Theta(f)$.

This allows us to get a nice effective sequence of recursive operators.

Proposition 21 *There exists an effective enumeration, $\Theta_0, \Theta_1, \cdots$ of recursive operators satisfying condition* (d) *above such that, for all recursive operators Θ, there exists an $i \in \mathbb{N}$ satisfying $\Theta(f) = \Theta_i(f)$ for all total functions f.*

Definition 22 (Freivalds *et al.* [13]). *Let $\mathcal{S}, \mathcal{C} \in \mathbf{Ex}$. Then $\mathcal{S}$ is called* **Ex**-reducible *to $\mathcal{C}$ (written: $\mathcal{S} \leq_{\mathbf{Ex}} \mathcal{C}$) iff there exist two recursive operators Θ and Ξ such that for all $f \in \mathcal{S}$,*

(a) $\Theta(f) \in \mathcal{C}$,
(b) *for any* **Ex**-*admissible sequence P for $\Theta(f)$, $\Xi(P)$ is* **Ex**-*admissible for f.*

Intuitively, if $\mathcal{S}$ is **Ex**-reducible to $\mathcal{C}$, then $\mathcal{C}$ is at least as difficult to **Ex**-learn as $\mathcal{S}$ is. Actually, if $\mathbf{M}$ **Ex**-learns $\mathcal{C}$, then $\mathcal{S}$ can be **Ex**-learned by a machine that, on any function $f \in \mathcal{S}$, outputs the sequence $\Xi(\mathbf{M}(\Theta(f)))$.

Definition 23 *Let $\mathbf{I}$ be a learning type and $\mathcal{C} \subseteq \mathcal{R}$. $\mathcal{C}$ is called* **Ex**-complete *in $\mathbf{I}$ iff $\mathcal{C} \in \mathbf{I}$, and for all $\mathcal{S} \in \mathbf{I}$, $\mathcal{S} \leq_{\mathbf{Ex}} \mathcal{C}$.*

Theorem 24 *Let $\mathcal{C} \in$* **WRefEx**. *Then there exists a class $\mathcal{S} \in$* **RefEx** *such that $\mathcal{S} \not\leq_{\mathbf{Ex}} \mathcal{C}$.*

Proof. Suppose $\mathcal{C} \in$ **WRefEx** as witnessed by $\mathbf{M}$. Note that for all recursive functions f, $\mathbf{M}(f)$ converges to a program for f, or converges to $\perp$. In particular, there is no recursive function f on which $\mathbf{M}$ makes infinitely many mind changes. Based on this we define a class $\mathcal{S} \in$ **RefEx** that is not **Ex**-reducible to $\mathcal{C}$. Let $\Theta_0, \Theta_1, \ldots$ be an enumeration of the operators as in Proposition 21. We will construct $\mathcal{S}$, with the following two properties:

(1) For each i, there exist distinct functions f, f' in $\mathcal{S}$ such that $\mathbf{M}(\Theta_i(f)) = \mathbf{M}(\Theta_i(f'))$. This would immediately imply that $\mathcal{S} \not\leq_{\mathbf{Ex}} \mathcal{C}$.
(2) $\mathcal{S} \in$ **RefEx**.

For each i, we will define some functions in $\mathcal{S}$, which have $f(0) = i$. These will be used to diagonalize against Θ_i (to satisfy (1) above).

For each i, do the following (independent staging construction for each i).

Let $\sigma_0 = \{(0, i)\}$. Go to stage 0.

Stage s
1. Put 0-ext(σ_s) and 1-ext(σ_s) in $\mathcal{S}$.
2. Let $f_0 =$ 0-ext(σ_s), and $f_1 =$ 1-ext(σ_s).
3. Search for t, if any, such that $\mathbf{M}(\Theta_i(f_0[t])) \neq \mathbf{M}(\Theta_i(f_1[t]))$.
4. If and when such a t is found, pick the least such t.
5. Suppose $\mathbf{M}(\Theta_i(f_w[t])) \neq \mathbf{M}(\Theta_i(\sigma_s))$, where $w \in \{0, 1\}$.
6. Let $\sigma_{s+1} = f_w[t]$.
7. Go to stage $s + 1$.

End stage s

Claim 25 *For each i, there are only finitely many stages.*

Proof. Otherwise $\mathbf{M}$ makes infinitely many mind changes on the recursive function $\Theta_i(\bigcup_{s\in\mathbb{N}} \sigma_s)$. □

For each i, one can effectively (in i) enumerate the initial segments which start with i, but are not extended by any function in $\mathcal{S}$. To see this, consider any $\tau \supseteq \{(0,i)\}$. Execute the stages as above, until

(i) a σ_s is defined such that $\tau \subseteq 0\text{-ext}(\sigma_s)$ or $\tau \subseteq 1\text{-ext}(\sigma_s)$ (in which case τ is extended by a function in $\mathcal{S}$), or

(ii) a stage s is reached such that σ_s is inconsistent with τ, and $\sigma_{s-1} \subseteq \tau$ (in which case τ is extended by a function in $\mathcal{S}$ iff $\tau \subseteq 0\text{-ext}(\sigma_{s-1})$ or $\tau \subseteq 1\text{-ext}(\sigma_{s-1})$), or

(iii) a stage s is reached such that $\sigma_s \subseteq \tau$, and, for $f_0 = 0\text{-ext}(\sigma_s)$ and $f_1 = 1\text{-ext}(\sigma_s)$, it is observed that $\mathbf{M}(\Theta_i(f_0[z])) = \mathbf{M}(\Theta_i(f_1[z]))$, for all $z \leq t$, and τ is inconsistent with both $f_0[t]$ and $f_1[t]$ (in which case τ is not extended by any function in $\mathcal{S}$).

Note that above exhausts all the possible cases by the construction of $\mathcal{S}$.

Moreover, for each i, there are only finitely many f such that $f(0) = i$ and $f \in \mathcal{S}$, and these f can be recursively enumerated (effectively in i). It follows that $\mathcal{S} \in \mathbf{RefEx}$.

Now by construction, for each i, (1) is satisfied, since for the last stage s which is executed, $\mathbf{M}(\Theta_i(0\text{-ext}(\sigma_s))) = \mathbf{M}(\Theta_i(1\text{-ext}(\sigma_s)))$, where both $0\text{-ext}(\sigma_s)$ and $1\text{-ext}(\sigma_s)$ are in $\mathcal{S}$. ∎

Theorem 24 immediately yields the following result.

Theorem 26 (1) *There is no* **Ex***-complete class in* **RefEx**.
(2) *There is no* **Ex***-complete class in* **WRefEx**.

In contrast to Theorem 26, **RelEx** contains an **Ex**-complete class.

Theorem 27 *There is an* **Ex***-complete class in* **RelEx**.

Proof. In [13] it was shown that *FINSUP* is **Ex**-complete in **Ex**. Moreover, *FINSUP* $\in$ **RelEx**, see proof of Theorem 16. Hence, *FINSUP* is **Ex**-complete in **RelEx**. ∎

We now present several characterizations for **RefEx**, **WRefEx** and **RelEx**. The first group of characterizations relates refutable learning to the established concept of *classification*. The main goal in recursion theoretic classification can be informally described as follows. Let be given some finite (or even infinite) family of function classes. Then, for an arbitrary function from the union of all these classes, one has to find out which of these classes the corresponding function belongs to, see [4,39,37,36,9]. What we need in our characterization theorems below will be some special cases of classification, namely classification where only *two* classes are involved in the classification process, more exactly, a class together with its complement; and semi-classification which is some weakening of classification. Notice that the corresponding characterizations using these kinds of classification are in a sense close to the definitions of learning refutably. Nevertheless, these characterizations are useful in that their characteristic conditions are easily testable, i.e. they allow to check, whether or not a given class is learnable with refutation. Furthermore, they also allow to create classes being learnable refutably in a given sense.

Let $\mathcal{R}_{0,?}$ denote the class of all computable functions mapping the set $\mathbb{N}$ into the set $\{0, ?\}$ and being everywhere defined.

Definition 28 *A class $\mathcal{S} \subseteq \mathcal{R}$ is called* finitely semi-classifiable *iff there is $c \in \mathcal{R}_{0,?}$ such that*

(a) *for every $f \in \mathcal{S}$, there is an $n \in \mathbb{N}$ such that $c(f[n]) = 0$,*
(b) *for every $f \in \overline{\mathcal{S}}$ and for all $n \in \mathbb{N}$, $c(f[n]) = ?$.*

Intuitively, a class $\mathcal{S} \subseteq \mathcal{R}$ is finitely semi-classifiable if for any function from that class, after some finite amount of time one finds out that the function belongs to the class, whereas for any other function, i.e. for any function from $\overline{\mathcal{S}}$, one finds out "nothing".

Theorem 29 *For any $\mathcal{C} \subseteq \mathcal{R}$, $\mathcal{C} \in$ **RefEx** iff $\mathcal{C}$ is contained in some class $\mathcal{S} \in$ **Ex** such that $\overline{\mathcal{S}}$ is finitely semi-classifiable.*

Proof. Necessity. Suppose $\mathcal{C} \in$ **RefEx** as witnessed by some total IIM $\mathbf{M}$. Let $\mathcal{S} = \mathbf{Ex}(\mathbf{M})$. Clearly, $\mathcal{C} \subseteq \mathcal{S}$. Furthermore, (i) for any $f \in \mathcal{S}$ and any $n \in \mathbb{N}$, $\mathbf{M}(f[n]) \neq \perp$, and (ii) for any $f \in \overline{\mathcal{S}}$, there is $n \in \mathbb{N}$ such that $\mathbf{M}(f[n]) = \perp$.

Now define c as follows.

$$c(f[n]) = \begin{cases} 0, & \text{if } \mathbf{M}(f[n]) = \perp; \\ ?, & \text{if } \mathbf{M}(f[n]) \neq \perp. \end{cases}$$

Clearly, $c \in \mathcal{R}_{0,?}$ and $\overline{\mathcal{S}}$ is finitely semi-classifiable by c.

Sufficiency. Suppose $\mathcal{C} \subseteq \mathcal{S} \subseteq \mathbf{Ex}(\mathbf{M})$, and $\overline{\mathcal{S}}$ is finitely semi-classifiable by some $c \in \mathcal{R}_{0,?}$. Now define $\mathbf{M}'$ as follows.

$$\mathbf{M}'(f[n]) = \begin{cases} \mathbf{M}(f[n]), & \text{if } c(f[n]) =?; \\ \perp, & \text{if } c(f[x]) = 0, \text{ for some } x \leq n. \end{cases}$$

It is easy to verify that $\mathbf{M}'$ **RefEx**-identifies $\mathcal{C}$. ∎

We can apply the characterization of **RefEx** above in order to show that **RefEx** contains "non-trivial" classes. Therefore, let

$$\mathcal{C} = \{f \mid f \in \mathcal{R} \ \& \ \varphi_{f(0)} = f \ \& \ (\forall x \in \mathbb{N})[\Phi_{f(0)}(x) \leq f(x+1)]\}.$$

Clearly, $\mathcal{C} \in \mathbf{Ex}$ and $\overline{\mathcal{C}}$ is finitely semi-classifiable. Hence, by Theorem 29, $\mathcal{C}$ is **RefEx**-learnable. Moreover, $\mathcal{C} \notin \mathbf{NUM}$ was shown in [40], Theorem 4.2. Hence, we get the following corollary illustrating that **RefEx** contains "algorithmically rich" classes, that is classes being not contained in any recursively enumerable class.

Corollary 30 $\mathbf{RefEx} \setminus \mathbf{NUM} \neq \emptyset$.

We now characterize **WRefEx**.

Definition 31 (Wiehagen and Smith [39]). *Let $\mathcal{C}, \mathcal{S} \subseteq \mathcal{R}$, where $\mathcal{C}, \mathcal{S}$ are disjoint. $(\mathcal{C}, \mathcal{S})$ is called* classifiable *iff there is $c \in \mathcal{R}_{0,1}$ such that for any $f \in \mathcal{C}$ and for almost all $n \in \mathbb{N}$, $c(f[n]) = 0$; and for any $f \in \mathcal{S}$ and for almost all $n \in \mathbb{N}$, $c(f[n]) = 1$.*

Thus, intuitively, a pair of disjoint classes $\mathcal{C}, \mathcal{S}$ is classifiable if for any function from the union of $\mathcal{C}$ and $\mathcal{S}$, in the limit one can find out which of the classes $\mathcal{C}$ or $\mathcal{S}$ this function belongs to. For characterizing **WRefEx**, we need a special case of classification, namely where the classes under consideration form a partition of the class of all recursive functions, i.e. one class is just the complement of the other.

Definition 32 *A class $\mathcal{C} \subseteq \mathcal{R}$ is called* classifiable *iff $(\mathcal{C}, \overline{\mathcal{C}})$ is classifiable.*

Theorem 33 *For any $\mathcal{C} \subseteq \mathcal{R}$, $\mathcal{C} \in \mathbf{WRefEx}$ iff $\mathcal{C}$ is contained in some classifiable class $\mathcal{S} \in \mathbf{Ex}$.*

Proof. Necessity. Suppose $\mathcal{C} \in \mathbf{WRefEx}$ as witnessed by some total IIM $\mathbf{M}$. Let $\mathcal{S} = \mathbf{Ex}(\mathbf{M})$. Clearly, $\mathcal{C} \subseteq \mathcal{S}$ and $\mathcal{S} \in \mathbf{Ex}$. Now define c as follows.

$$c(f[n]) = \begin{cases} 0, & \text{if } \mathbf{M}(f[n]) \neq \perp; \\ 1, & \text{if } \mathbf{M}(f[n]) = \perp. \end{cases}$$

Then, clearly, $\mathcal{S}$ is classifiable by c.

Sufficiency. Suppose $\mathcal{C} \subseteq \mathcal{S} \subseteq \mathbf{Ex}(\mathbf{M})$, and let $\mathcal{S}$ be classifiable by some $c \in \mathcal{R}_{0,1}$. Then, define $\mathbf{M}'$ as follows.

$$\mathbf{M}'(f[n]) = \begin{cases} \mathbf{M}(f[n]), & \text{if } c(f[n]) = 0 \\ \perp, & \text{if } c(f[n]) = 1. \end{cases}$$

Clearly, $\mathbf{M}'$ witnesses that $\mathcal{C} \in \mathbf{WRefEx}$. ∎

Finally, we give a characterization of **RelEx** in terms of semi-classifiability.

Definition 34 (Stephan [37]). *A class $\mathcal{S} \subseteq \mathcal{R}$ is called* semi-classifiable *iff there is $c \in \mathcal{R}_{0,?}$ such that*

(a) *for any $f \in \mathcal{S}$ and almost all $n \in \mathbb{N}$, $c(f[n]) = 0$,*
(b) *for any $f \in \overline{\mathcal{S}}$ and infinitely many $n \in \mathbb{N}$, $c(f[n]) = ?$.*

Intuitively, a class of recursive functions is semi-classifiable if for any function from this class, in the limit one is able to find out that this function belongs to the class, while for any recursive function outside that class, there is no required evidence in the limit to know where this function comes from.

Theorem 35 *For any $\mathcal{C} \subseteq \mathcal{R}$, $\mathcal{C} \in \mathbf{RelEx}$ iff $\mathcal{C}$ is contained in some semi-classifiable class $\mathcal{S} \in \mathbf{Ex}$.*

Proof. Necessity. Suppose $\mathcal{C} \in \mathbf{RelEx}$ by some total IIM $\mathbf{M}$. Let $\mathcal{S} = \mathbf{Ex}(\mathbf{M})$. Clearly, $\mathcal{C} \subseteq \mathcal{S}$. In order to show that $\mathcal{S}$ is semi-classifiable, define c as follows.

$$c(f[n]) = \begin{cases} 0, & \text{if } n = 0 \text{ or } \mathbf{M}(f[n-1]) = \mathbf{M}(f[n]); \\ ?, & \text{if } n > 0 \text{ and } \mathbf{M}(f[n-1]) \neq \mathbf{M}(f[n]). \end{cases}$$

Now, for any $f \in \mathcal{S}$, $\mathbf{M}(f)\downarrow$, and thus $c(f[n]) = 0$ for almost all $n \in \mathbb{N}$. On the other hand, if $f \in \overline{\mathcal{S}}$ then $f \notin \mathbf{Ex}(\mathbf{M})$. Consequently, since $\mathbf{M}$ is reliable and total, we have $\mathbf{M}(f[n-1]) \neq \mathbf{M}(f[n])$ for infinitely many $n \in \mathbb{N}$. Hence $c(f[n]) = ?$ for infinitely many n. Thus, $\mathcal{S}$ is semi-classifiable by c.

Sufficiency. Suppose $\mathcal{C} \subseteq \mathcal{S} \subseteq \mathbf{Ex}(\mathbf{M})$. Suppose $\mathcal{S}$ be semi-classifiable by some $c \in \mathcal{R}_{0,?}$. Define $\mathbf{M}'$ as follows.

$$\mathbf{M}'(f[n]) = \begin{cases} \mathbf{M}(f[n]), & \text{if } c(f[n]) = 0; \\ n, & \text{if } c(f[n]) = ?. \end{cases}$$

Clearly, for any $f \in \mathcal{S}$, for almost all n, $c(f[n]) = 0$. Hence $\mathbf{M}'$ will **Ex**-identify f, since $\mathbf{M}$ does so. If $f \in \overline{\mathcal{S}}$, then $c(f[n]) = ?$ for infinitely many n. Consequently, $\mathbf{M}'$ diverges on f caused by arbitrarily large outputs. Thus, $\mathbf{M}'$ **RelEx**-identifies $\mathcal{C}$. ∎

Notice that there is some kind of "dualism" in the characterizations of **RefEx** and **RelEx** above. Indeed, a class is **RefEx**-learnable in case this class is contained in some **Ex**-learnable class the *complement* of which is finitely semi-classifiable. In contrast, a class is **RelEx**-learnable if this class is contained in an **Ex**-learnable class that *itself* is semi-classifiable.

The characterizations of the second group, this time for **RefEx** and **RelEx**, significantly differ from the characterizations presented above in two points. First, the characteristic conditions are stated here in terms that formally have nothing to do with learning. And second, the sufficiency proofs being constructively again make clear where the "refuting ability" of the corresponding learning machines comes from in general. For stating the corresponding characterization of **RefEx**, we need the following notions.

Definition 36 *A numbering ψ is called* strongly one-to-one *iff there is a recursive function d of two arguments such that for any distinct $i, j \in \mathbb{N}$, there exists an $x < d(i,j)$ such that $\psi_i(x) \neq \psi_j(x)$.*

Obviously, any strongly one-to-one numbering is one-to-one. Moreover, given any distinct ψ-indices i and j, the functions ψ_i and ψ_j do not only differ, but one can effectively compute a bound on the least argument on which these functions differ.

Definition 37 (Rice [34]). *A class $\Pi \subseteq \mathcal{P}$ is called* completely r.e. *iff $\{i \mid \varphi_i \in \Pi\}$ is recursively enumerable.*

Thus, a class of partial recursive functions is completely r.e. if its complete index set, i.e. the set of all programs of functions from Π in the acceptable programming system φ, is recursively enumerable. We will need the following characterization of completely r.e. classes.

Lemma 38 ([34,27]). *For any $\Pi \subseteq \mathcal{P}$, Π is completely r.e. iff there is an r.e. subset S of* SEG *such that $\Pi = \{g \mid g \in \mathcal{P} \;\&\; (\exists\, \sigma \in S)[\sigma \subseteq g]\}$.*

Hence, a class Π is completely r.e. if for some effective class of *finite* functions, Π contains exactly all the computable superfunctions of these finite functions.

Theorem 39 *For any $\mathcal{C} \subseteq \mathcal{R}$, $\mathcal{C} \in$ **RefEx** iff there are numberings ψ and ϱ such that*

(1) *ψ is strongly one-to-one and $\mathcal{C} \subseteq \mathcal{P}_\psi$,*

(2) *$\mathcal{P}_\varrho$ is completely r.e. and $\mathcal{R}_\varrho = \overline{\mathcal{R}_\psi}$.*

Proof. Necessity. Without loss of generality assume that $\mathcal{C}$ is infinite. Let $\mathcal{C} \in \mathbf{RefEx}$ as witnessed by a total IIM $\mathbf{M}$. Then let

$$Z = \{(z,n) \mid (\forall x < n)[\varphi_z(x)\downarrow] \;\&\; \mathbf{M}(\varphi_z[n]) = z \;\&\; [n = 0 \vee \mathbf{M}(\varphi_z[n-1]) \neq \mathbf{M}(\varphi_z[n])]\}.$$

Intuitively, the set Z contains "initial segments" of any function where $\mathbf{M}$ might begin to converge to a correct hypothesis. Let e be a 1–1 recursive function such that $Z = \text{range}(e)$. For any $i, j, x \in N$, where $e(i) = (z,n)$ and $e(j) = (w,m)$, define

$$\psi_i(x) = \begin{cases} \varphi_z(x), & \text{if } x < n; \\ \varphi_z(x), & \text{if } x \geq n \text{ and } (\forall y \leq x)[\varphi_z(y)\downarrow] \text{ and} \\ & (\forall y \mid n < y \leq x+1)[\mathbf{M}(\varphi_z[y]) = \mathbf{M}(\varphi_z[n])]; \\ \uparrow, & \text{otherwise.} \end{cases}$$

and $d(i,j) = \max(\{n,m\})$.

Then it can easily be seen that ψ is a strongly one-to-one numbering as witnessed by the function d.
In order to show $\mathcal{C} \subseteq \mathcal{P}_\psi$ we prove a somewhat stronger result which is needed in the following. Therefore, let $\mathcal{S}$ denote the class of *all* recursive functions that are **Ex**-learnable by $\mathbf{M}$. Clearly, $\mathcal{C} \subseteq \mathcal{S}$.

Claim 40 *$\mathcal{S} = \mathcal{R}_\psi$.*

Proof. Let $f \in \mathcal{S}$. Then there is a minimal $n \in \mathbb{N}$ and a $z \in \mathbb{N}$ such that $\varphi_z = f$ and for all $m \geq n$, $\mathbf{M}(f[m]) = z$. Consequently, $(z,n) \in Z$ and $\psi_i = \varphi_z = f$, where $e(i) = (z,n)$. Hence $f \in \mathcal{R}_\psi$.
Let now $f \in \mathcal{R}_\psi$. Let $f = \psi_i$ and $e(i) = (z,n)$. Then $\psi_i = \varphi_z$, since $\psi_i = f$ is everywhere defined. Consequently, for all $m \geq n, \mathbf{M}(f[m]) = \mathbf{M}(\varphi_z[m]) = \mathbf{M}(\varphi_z[n]) = z$. Hence f is **Ex**-learnable by $\mathbf{M}$, and thus $f \in \mathcal{S}$. □

Claim 40 above completes the proof of condition (1).
In order to show condition (2) let

$$A = \{\sigma \in \text{SEG} \mid \mathbf{M}(\sigma) = \perp\}.$$

Clearly, A is recursively enumerable.

Finally, let ϱ be an arbitrary numbering such that

$$\mathcal{P}_\varrho = \{\eta \mid \eta \in \mathcal{P} \;\&\; (\exists \sigma \in A)[\sigma \subseteq \eta]\}.$$

Obviously, $\mathcal{P}_\varrho$ is completely r.e. by Lemma 38.

Claim 41 $\mathcal{R}_\varrho = \overline{\mathcal{R}_\psi}$.

Proof. By Claim 40, it suffices to prove that $\mathcal{R}_\varrho = \overline{\mathcal{S}}$.

Let $f \in \mathcal{R}_\varrho$. Then, by the definitions of $\mathcal{P}_\varrho$ and A, $\mathbf{M}(f[n]) = \perp$ for some n. Consequently, f cannot be **Ex**-learned by $\mathbf{M}$. Hence $f \in \overline{\mathcal{S}}$.

Suppose now $f \in \overline{\mathcal{S}}$. Then, by definition of **RefEx**, f must be refuted by $\mathbf{M}$. Thus, $\mathbf{M}(f[n]) = \perp$ for some n, and hence $f \in \mathcal{P}_\varrho$. □

Claim 41 completes the proof of condition (2).

Sufficiency. Let ψ be a strongly one-to-one numbering as witnessed by a corresponding function d. Let ϱ be a numbering such that for some r.e. $S \subseteq \mathrm{SEG}$, $\mathcal{P}_\varrho = \{\eta \mid \eta \in \mathcal{P} \ \&\ (\exists \sigma \in S)[\sigma \subseteq \eta]\}$, and $\mathcal{R}_\varrho = \overline{\mathcal{R}_\psi}$. Then an IIM $\mathbf{M}$ that **RefEx**-learns $\mathcal{R}_\psi \supseteq \mathcal{C}$ can be defined as follows. Let $f \in \mathcal{R}$.

$\mathbf{M}(f) =$ " In parallel do both (A) and (B).
(A) Go to stage 0.
Stage i.
Output i. Check if there is $j \neq i$ such that
$f[d(i,j)] \subseteq \psi_j$
in which case go to stage $i+1$.
End Stage i.
(B) Check if there is $\sigma \in S$ such that$\sigma \subseteq f$
in which case output $\perp$ forever."

Claim 42 $\mathbf{M}$ **Ex**-*learns any function from* $\mathcal{R}_\psi$.

Proof. Let $f \in \mathcal{R}_\psi$. First notice that (B) can never happen, since otherwise $f \in \mathcal{R}_\varrho$ would follow, a contradiction to $\mathcal{R}_\varrho$ and $\mathcal{R}_\psi$ being disjoint. Let $f = \psi_z$. Then, clearly, for any $i < z$, $f[d(i,z)] \subseteq \psi_z$ will hold. Hence, in (A), stage z will be reached. But stage z can never be left, since this would yield $f[d(z,j)] \subseteq \psi_j$ for some $j \neq z$, implying the contradiction $\psi_z \neq f$ via $[(\exists x < d(z,j))[\psi_z(x) \neq \psi_j(x)]]$. Consequently, on f, $\mathbf{M}$ will converge to z, thus **Ex**-learning f. □

Claim 43 $\mathbf{M}$ *refutes any function from* $\overline{\mathcal{R}_\psi}$.

Proof. For any function $f \in \overline{\mathcal{R}_\psi} = \mathcal{R}_\varrho$, (B) happens by the definition of $\mathcal{P}_\varrho$. Hence $\mathbf{M}$ refutes f. □

Claims 42 and 43 complete the sufficiency proof. ∎

It follows from the proof of Theorem 39 that in **RefEx**-learning the processes of learning and refuting, respectively, can be nicely separated. Actually, in general, a suitable machine can be provided with two spaces, one for learning, ψ, and one for refuting, ϱ. If and when the "search for refutation" in the refutation space has been successful, then the learning process can be stopped forever. This search for refutation is based on the property that the whole space for refutation forms a completely r.e. class $\mathcal{P}_\varrho$ of partial recursive functions. Then, by the characterization lemma for completely r.e. classes, any system S of "representatives" for $\mathcal{P}_\varrho$ can serve as a set of indicators for refutation. The spaces for learning and for refuting are interconnected by the essential property that their recursive kernels, $\mathcal{R}_\psi$ and $\mathcal{R}_\varrho$, disjointly exhaust the class of *all* recursive functions. This property eventually guarantees that each recursive function either will be learned or refuted. Notice that the above characterization of **RefEx** is in a sense "more granular" than the characterization of **RefEx** by Theorem 29. Intuitively, the characterization of Theorem 29 requires that one should be able to find out *anyhow* if the given function does not belong to the class to be learned. The characterization of Theorem 39 now makes precise *how* this task can be done. Indeed, the set S may be thought as just sampling all possibilities of violating the structure of the functions from the class to be learned, thereby indicating if and when the corresponding function has to be refuted. Furthermore, notice that the **RefEx**-characterization of Theorem 39 is incremental to a characterization of **Ex** in that the existence of a numbering with condition (1) above is just necessary and sufficient for **Ex**-learning the corresponding class $\mathcal{C}$, see [38]. Finally, notice that the refutation space could be "economized" in the same strict manner as the learning space, that is, it can be made one-to-one.

The following characterization of **RelEx** is a slight modification of a result from [21].

Theorem 44 *For every $\mathcal{C} \subseteq \mathcal{R}$, $\mathcal{C} \in$ **RelEx** iff there are a numbering ψ and a function $d \in \mathcal{R}$ such that*

(1) *for all $f \in \mathcal{R}$, if $H_f = \{i \mid f[d(i)] \subseteq \psi_i\}$ is finite, then H_f contains a ψ-index of f,*
(2) *for every $f \in \mathcal{C}$, H_f is finite.*

Proof. Necessity. Let $\mathcal{C} \in$ **RelEx** as witnessed by some total IIM $\mathbf{M}$, and let

$$Y = \{f[n] \mid f \in \mathcal{R} \;\&\; [n = 0 \;\vee\; \mathbf{M}(f[n-1]) \neq \mathbf{M}(f[n])]\}.$$

Suppose e is a 1–1 recursive function such that $\mathrm{range}(e) = Y$. For $i, x \in \mathbb{N}$ and $e(i) = f[n]$, let $d(i) = n$ and

$$\psi_i(x) = \begin{cases} f(x), & \text{if } x < n; \\ \varphi_{\mathbf{M}(f[n])}(x), & \text{otherwise.} \end{cases}$$

Clearly, ψ is a numbering and $d \in \mathcal{R}$. Now let $f \in \mathcal{R}$ be such that $H_f = \{i_0, \ldots, i_m\}$ is finite. Notice that H_f must be non-empty, since $f[0] \in Y$ and, hence, by definition of ψ, $i_0 \in H_f$, where $e(i_0) = f[0]$. For $j \leq m$, let $\alpha_j = e(i_j)$. Without loss of generality, let m be such that $|\alpha_m|$ is maximum among $|\alpha_j|$ for $j \leq m$. Then, by definition of Y and ψ, $\mathbf{M}(f[n]) = \mathbf{M}(\alpha_m)$, for any $n \geq |\alpha_m|$. Hence $\mathbf{M}$ converges on f. Since $\mathbf{M}$ is reliable, $f = \varphi_{\mathbf{M}(\alpha_m)}$ follows. Moreover, by the definition of ψ, we have $\psi_{i_m} = f$. Consequently, H_f contains a ψ-index of f. This proves condition (1). For showing condition (2) suppose $f \in \mathcal{C}$. Then $\mathbf{M}$ converges on f. Consequently, there are at most finitely many $n \in \mathbb{N}$ such that $f[n] \in Y$. By definition of ψ, this implies that H_f is finite.

Sufficiency. Informally, an IIM reliably learning $\mathcal{C}$, on every function $f \in \mathcal{R}$, searches for all the elements of the set H_f and applies the amalgamation technique, see [10], to this set. In order to proceed more formally, let $c \in \mathcal{R}$ be such that for any $i \in \mathbb{N}$, $\psi_i = \varphi_{c(i)}$. Let amal be a recursive function mapping any finite set I of ψ-indices to a φ-index such that for any $x \in \mathbb{N}$, $\varphi_{\mathrm{amal}(I)}(x)$ is defined by running $\varphi_{c(i)}(x)$ for every $i \in I$ in parallel and taking the first value obtained, if any. For any $f \in \mathcal{R}$ and $n \in \mathbb{N}$, let

$$H_{f,n} = \{i \mid i \leq n \;\&\; d(i) \leq n \;\&\; (\forall x < d(i))[\Phi_{c(i)}(x) \leq n \;\&\; \varphi_{c(i)}(x) = f(x)]\}.$$

Intuitively, $H_{f,n}$ is the set of all ψ-indices i such that $i \in H_f$ can be verified within a uniformly (in n) bounded number of computation steps. Let

$$H^+_{f,n} = \{i \mid i \in H_{f,n} \;\&\; (\forall x < n)[\Phi_{c(i)}(x) \leq n \Rightarrow \varphi_{c(i)}(x) = f(x)]\}.$$

Thus, $H^+_{f,n}$ is the subset of $H_{f,n}$ consisting of all indices i such that on any argument less than n, ψ_i does not contradict f within n steps of computation. Finally, let $H_{f,-1} = \emptyset$.

Then define an IIM $\mathbf{M}$ as follows.

$$\begin{aligned}\mathbf{M}(f[n]) = \text{“}\quad & \text{If } H_{f,n} = H_{f,n-1} \neq \emptyset, \text{ then output } \mathrm{amal}(H^+_{f,n}).\\ & \text{If } H_{f,n} = \emptyset \text{ or } H_{f,n} \neq H_{f,n-1}, \text{ then output } n.\text{”}\end{aligned}$$

Claim 45 *For any $f \in \mathcal{R}$, if H_f is finite, then* $\mathbf{M}$ **Ex**-*identifies f.*

Proof. By assumption of the claim we can conclude $\lim_{n\to\infty} H_{f,n} = H_f$. Therefore, $H^+_f = \lim_{n\to\infty} H^+_{f,n}$ exists, and H^+_f contains exactly every $i \in H_f$ such that ψ_i is a subfunction of f, including some ψ-index of f, by condition (1). Clearly, $\mathbf{M}(f[n])$ converges to $j = \mathrm{amal}(H^+_f)$, and $\varphi_j = f$. □

By Claim 45 and condition (2), $\mathbf{M}$ identifies $\mathcal{C}$.

It remains to show that $\mathbf{M}$ works reliably.

Claim 46 *For any $f \in \mathcal{R}$, if $\mathbf{M}$ converges on f, then $\mathbf{M}$ $\mathbf{Ex}$-identifies f.*

Proof. By Claim 45, it suffices to prove that if $\mathbf{M}$ converges on f, then H_f is finite. Suppose to the contrary that H_f is infinite. Then, by definition of $\mathbf{M}$, $\mathbf{M}$ *diverges* on f, a contradiction. □

This completes the proof of sufficiency. ∎

Theorem 44 instructively clarifies where the ability to learn reliably may come from. Mainly, it comes from the properties of a well-chosen space of hypotheses. In any such space ψ exhibited by Theorem 44, for any function f from the class to be learned, there are only finitely many "candidates" for ψ-indices of f, the set H_f. This *finiteness* of H_f together with the fact that H_f then contains a ψ-index of f, make sure that the amalgamation technique succeeds in learning any such f. Conversely, the *infinity* of this set H_f of candidates automatically ensures that the learning machine as defined in the sufficiency proof of Theorem 44 *diverges* on f. This is achieved by causing the corresponding machine to output arbitrarily large hypotheses on every function $f \in \mathcal{R}$ with H_f being infinite.

4 $\mathbf{Ex}^a$-Learning and $\mathbf{Bc}^a$-Learning Refutably

In this section, we consider $\mathbf{Ex}$-learning and $\mathbf{Bc}$-learning *with anomalies* refutably. Again, we will derive both strengths and weaknesses of refutable learning. As it turns out, many results of standard learning, i.e. without refutation, stand refutably. Specifically, this yields several hierarchies for refutable learning. Furthermore, we show that in general one cannot trade the strictness of the refutability constraints for the liberality of the learning criteria.

For $\eta, \eta' \in \mathcal{P}$ and $a \in \mathbb{N}$, we write $\eta =^a \eta'$ and $\eta =^* \eta'$ iff $\mathrm{card}(\{x \mid \eta(x) \neq \eta'(x)\}) \leq a$ and $\mathrm{card}(\{x \mid \eta(x) \neq \eta'(x)\}) < \infty$, respectively.

Definition 47 ([15,5,10]). *Let $a \in \mathbb{N} \cup \{*\}$, let $f \in \mathcal{R}$ and let $\mathbf{M}$ be an IIM.*

(a) $\mathbf{M}$ $\mathbf{Ex}^a$-identifies f *(written: $f \in \mathbf{Ex}^a(\mathbf{M})$) just in case, there exists an i such that $\mathbf{M}(f)\downarrow = i$ and $\varphi_i =^a f$.*
(b) $\mathbf{M}$ $\mathbf{Ex}^a$-identifies $\mathcal{C}$ *iff $\mathbf{M}$ $\mathbf{Ex}^a$-identifies each $f \in \mathcal{C}$.*
(c) $\mathbf{Ex}^a = \{\mathcal{C} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{C} \subseteq \mathbf{Ex}^a(\mathbf{M})]\}$.

Thus, in $\mathbf{Ex}^a$-learning the final hypothesis may be slightly incorrect in that it

is allowed to contain at most a anomalies. Note that $\mathbf{Ex} = \mathbf{Ex}^0$.

Definition 48 ([2,10]). *Let $a \in \mathbb{N} \cup \{*\}$, let $f \in \mathcal{R}$ and let $\mathbf{M}$ be an IIM.*

(a) $\mathbf{M}$ $\mathbf{Bc}^a$-identifies f *(written: $f \in \mathbf{Bc}^a(\mathbf{M})$) iff, for all but finitely many $n \in \mathbb{N}$, $\varphi_{\mathbf{M}(f[n])} =^a f$.*
(b) $\mathbf{M}$ $\mathbf{Bc}^a$-identifies $\mathcal{C}$ *iff* $\mathbf{M}$ $\mathbf{Bc}^a$*-identifies each $f \in \mathcal{C}$.*
(c) $\mathbf{Bc}^a = \{\mathcal{C} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{C} \subseteq \mathbf{Bc}^a(\mathbf{M})]\}$.

Note that $\mathbf{Bc} = \mathbf{Bc}^0$.

Harrington [10] showed that $\mathcal{R} \in \mathbf{Bc}^*$. Thus we will mostly consider only $\mathbf{Bc}^a$ for $a \in \mathbb{N}$, in the following.

We can now define $\mathbf{IEx}^a$ and $\mathbf{IBc}^a$ for $\mathbf{I} \in \{\mathbf{Ref}, \mathbf{WRef}, \mathbf{Rel}\}$ analogously to Definitions 5, 6, and 7. We only give the definitions of $\mathbf{RefEx}^a$ and $\mathbf{RelBc}^a$ as examples.

Definition 49 *Let $a \in \mathbb{N} \cup \{*\}$ and let $\mathbf{M}$ be an IIM.* $\mathbf{M}$ $\mathbf{RefEx}^a$-identifies $\mathcal{C}$ *iff*

(a) $\mathcal{C} \subseteq \mathbf{Ex}^a(\mathbf{M})$.
(b) *For all $f \in \mathbf{Ex}^a(\mathbf{M})$, for all n, $\mathbf{M}(f[n]) \neq \perp$.*
(c) *For all $f \in \mathcal{R}$ such that $f \notin \mathbf{Ex}^a(\mathbf{M})$, there exists an n such that $(\forall m < n)[\mathbf{M}(f[m]) \neq \perp]$ and $(\forall m \geq n)[\mathbf{M}(f[m]) = \perp]$.*

Definition 50 (Kinber and Zeugmann [23]). *Let $a \in \mathbb{N} \cup \{*\}$ and let $\mathbf{M}$ be an IIM.* $\mathbf{M}$ $\mathbf{RelBc}^a$-identifies $\mathcal{C}$ *iff*

(a) $\mathcal{C} \subseteq \mathbf{Bc}^a(\mathbf{M})$.
(b) *For all $f \in \mathcal{R}$ such that $f \notin \mathbf{Bc}^a(\mathbf{M})$, there exist infinitely many n such that $\mathbf{M}(f[n]) = \perp$.*

Note that the learning types $\mathbf{RelEx}^a$ and $\mathbf{RelBc}^a$ were studied firstly in [22] and [23], respectively.

Our first result points out some weakness of learning refutably. It shows that there are classes which, on the one hand, are easy to learn in the standard sense of $\mathbf{Ex}$-learning without any mind change, but, on the other hand, which are not learnable refutably, even if we allow both the most liberal type of learning refutably, namely reliable learning, and the very rich type of $\mathbf{Bc}$-learning with an arbitrarily large number of anomalies. For proving this result, we need the following proposition.

Proposition 51 (a) *For any $a \in \mathbb{N}$ and any $\sigma \in \mathrm{SEG}$, $\{f \in \mathcal{R} \mid \sigma \subseteq f\} \notin \mathbf{Bc}^a$.*
(b) *For any $a \in \mathbb{N}$ and any $\sigma \in \mathrm{SEG}_{0,1}$, $\{f \in \mathcal{R}_{0,1} \mid \sigma \subseteq f\} \notin \mathbf{Bc}^a$.*

Proof. We only show part (a). Part (b) can be shown similarly. Suppose by way of contradiction that $a \in \mathbb{N}$ and $\sigma \in \mathrm{SEG}$ are such that $\{f \in \mathcal{R} \mid \sigma \subseteq f\} \in \mathbf{Bc}^a$. Suppose $\mathbf{M}$ $\mathbf{Bc}^a$-identifies $\{f \in \mathcal{R} \mid \sigma \subseteq f\}$. Let g_f be defined as follows:

$$g_f(x) = \begin{cases} \sigma(x), & \text{if } x < |\sigma|; \\ f(x - |\sigma|), & \text{otherwise.} \end{cases}$$

Let $g_{f[n]} = g_f[n + |\sigma|]$.

Let *prog* be a recursive function such that $\varphi_{prog(p)}(x) = \varphi_p(x + |\sigma|)$.

Define $\mathbf{M}'$ as follows: $\mathbf{M}'(f[n]) = prog(\mathbf{M}(g_{f[n]}))$. It is easy to verify that if $\mathbf{M}$ $\mathbf{Bc}^a$-identifies g_f, then $\mathbf{M}'$ $\mathbf{Bc}^a$-identifies f. It follows that $\mathbf{M}'$ $\mathbf{Bc}^a$-identifies $\mathcal{R}$. This contradicts the $\mathbf{Bc}^a$-hierarchy theorem in [10], and thus the proposition follows. ∎

Next, we define **Ex**-learning without mind changes, or, equivalently, *finite learning*. Informally, here the learning machine has "one shot" only to do its learning task.

Definition 52 (Gold [15]). *Let $f \in \mathcal{R}$ and let $\mathbf{M}$ be an IIM.*

(a) $\mathbf{M}$ **Fin**-identifies f *(written: $f \in \mathbf{Fin}(\mathbf{M})$) iff there is $n \in \mathbb{N}$ such that for any $x < n$, $\mathbf{M}(f[x]) = ?$, $\mathbf{M}(f[n]) \in \mathbb{N}$, and $\varphi_{\mathbf{M}(f[n])} = f$.*
(b) $\mathbf{M}$ **Fin**-identifies $\mathcal{C}$ *iff $\mathbf{M}$* **Fin***-identifies each $f \in \mathcal{C}$.*
(c) $\mathbf{Fin} = \{\mathcal{C} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{C} \subseteq \mathbf{Fin}(\mathbf{M})\}$.

Theorem 53 *For all $a \in \mathbb{N}$, $\mathbf{Fin} \setminus \mathbf{RelBc}^a \neq \emptyset$.*

Proof. Let $\mathcal{C} = \{f \in \mathcal{R}_{0,1} \mid f \neq \text{Zero} \ \& \ \varphi_{\min(\{x \mid f(x)=1\})} = f\}$. Clearly, $\mathcal{C} \in \mathbf{Fin}$. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{RelBc}^a$-identifies $\mathcal{C}$. Then, by Kleene recursion theorem [35], there exists an e such that φ_e may be defined in stages as follows. Let $\varphi_e(x) = 0$, for $x < e$, and $\varphi_e(e) = 1$.

Let φ_e^s denote φ_e defined before stage s. Go to stage 0.

Stage s.
1. Search for a $\tau \in \mathrm{SEG}_{0,1}$ properly extending φ_e^s such that $\mathbf{M}(\tau) = \perp$.
2. If and when such a τ is found, let $\varphi_e^{s+1} = \tau$, and go to stage $s + 1$.

End stage s

We consider two cases.

Case 1. All stages finish (i.e. step 1 succeeds in all stages).

In this case $\varphi_e \in \mathcal{R} \cap \mathcal{C}$, and $\mathbf{M}$ outputs $\perp$ on infinitely many initial segments of φ_e.

Case 2. Stage s starts but does not finish.

In this case, by definition of $\mathbf{RelBc}^a$, $\mathbf{M}$ must $\mathbf{Bc}^a$-identify $\{f \in \mathcal{R}_{0,1} \mid \varphi_e^s \subseteq f\}$, a contradiction to Proposition 51.

From the above cases it follows that $\mathbf{M}$ does not $\mathbf{RelBc}^a$-identify $\mathcal{C}$. ∎

Next we show that allowing anomalies can help in learning refutably. Indeed, while $\mathbf{Ex}^{a+1} \setminus \mathbf{Ex}^a \neq \emptyset$ was shown in [10], we now strengthen this result to $\mathbf{RefEx}$-learning with anomalies. Therefore, we need the following lemma.

Lemma 54 *For every $a \in \mathbb{N}$, there exists a function $p \in \mathcal{R}$ such that, for all $i \in \mathbb{N}$:*

(a) $\text{range}(\varphi_{p(i)}) \subseteq \{0, 1\}$.

(b) $\varphi_{p(i)}$ *is undefined on at most $a+1$ inputs.*

(c) $\varphi_{p(i)}(i) = 1$, *and, for all $x < i$, $\varphi_{p(i)}(x) = 0$.*

(d) $\{f \in \mathcal{R}_{0,1} \mid \varphi_{p(i)} \subseteq f\} \not\subseteq \mathbf{Ex}^a(\mathbf{M}_i)$.

Proof. The lemma is proved by a modification of the proof of $\mathbf{Ex}^{a+1} \setminus \mathbf{Ex}^a \neq \emptyset$ in [10]. By the parameterized recursion theorem [35], there exists a recursive function p such that $\varphi_{p(i)}$ may be defined in stages as follows. $\varphi_{p(i)}(i) = 1$, and $\varphi_{p(i)}(x) = 0$, for $x < i$. Let x_s denote the least x such that $\varphi_{p(i)}(x)$ has not been defined before stage s.

Go to stage 0.

Stage s

1. Dovetail steps 2 and 3, until step 2 succeeds. If and when step 2 succeeds, go to step 4.
2. Search for a $\tau \in \text{SEG}_{0,1}$ such that
   $\tau(x) = \varphi_{p(i)}(x)$, for $x < x_s$,
   $\tau(x)\downarrow$, for $x_s \leq x \leq x_s + a$,
   $\tau(x) = 0$ or undefined, for $x > x_s + a$,
   and $\mathbf{M}_i(\tau) \neq \mathbf{M}_i(\varphi_{p(i)}[x_s])$.
3. For $x = x_s + a + 1$ to ∞
   Let $\varphi_{p(i)}(x) = 0$.
   EndFor
4. If and when such a τ is found let,

5. $\varphi_{p(i)}(x) = \tau(x)$, for $x < |\tau|$ such that $\varphi_{p(i)}(x)$ has not been defined upto now.
6. Go to stage $s+1$.

End stage s

Fix i. (a) and (c) clearly hold. We now consider two cases.

Case 1. All stages terminate.

In this case $\varphi_{p(i)}$ is total. Thus (b) is satisfied. Also due to step 5, $\mathbf{M}_i$ changes its mind infinitely often on $\varphi_{p(i)}$.

Case 2. Stage s starts but does not terminate.

In this case $\varphi_{p(i)}$ is undefined only on $\{x \mid x_s \leq x \leq x_s + a\}$, thus (b) is satisfied. Also, for all $f \in \mathcal{R}_{0,1}$ such that $\varphi_{p(i)} \subseteq f$, $\mathbf{M}_i(f) = \mathbf{M}_i(\varphi_{p(i)}[x_s])$. Let $e = \mathbf{M}_i(\varphi_{p(i)}[x_s])$. Let g be defined as follows:

$$g(x) = \begin{cases} \varphi_{p(i)}(x), & \text{if } x < x_s \text{ or } x > x_s + a; \\ 0, & \text{if } x_s \leq x \leq x_s + a, \text{ and } \varphi_e(x)\uparrow; \\ 1 \dot{-} \varphi_e(x), & \text{otherwise.} \end{cases}$$

It is easy to verify that $g \in \mathcal{R}_{0,1}$, $g \supseteq \varphi_{p(i)}$, and $\mathbf{M}_i(g)\downarrow = \mathbf{M}_i(\varphi_{p(i)}[x_s]) = e$. However, $\varphi_e \neq^a g$, since $g(x) \neq \varphi_e(x)$, for $x_s \leq x \leq x_s + a$. Thus (d) holds.

Lemma follows from above cases. ∎

Theorem 55 *For all $a \in \mathbb{N}$, $\mathbf{RefEx}^{a+1} \setminus \mathbf{Ex}^a \neq \emptyset$.*

Proof. Let p be as in Lemma 54. Let $\mathcal{C}_i = \{\text{Zero}\} \cup \{f \in \mathcal{R}_{0,1} \mid \varphi_{p(i)} \subseteq f\}$. Let $\mathcal{C} = \bigcup_{i \in \mathbb{N}} \mathcal{C}_i$. By Lemma 54 (d), it follows that $\mathcal{C} \notin \mathbf{Ex}^a$.

Now define $\mathbf{M}$ as follows. Let z be a program for Zero. Let $\text{MinO}(\sigma) = \min(\{x \mid \sigma(x) = 1\})$,

$\mathbf{M}(\sigma)$
1. If $\sigma \subseteq$ Zero, then output z.
   Else, let $i = \text{MinO}(\sigma)$.
2. If there exists an x such that $\varphi_{p(i)}(x)$ converges in at most $|\sigma|$ steps, and $\varphi_{p(i)}(x) \neq \sigma(x)$, then output $\perp$.
   Else output $p(i)$.

End $\mathbf{M}(\sigma)$

Clearly, $\mathbf{M}\ \mathbf{Ex}^{a+1}$-identifies $\mathcal{C}$. If $f \notin \mathcal{C}$, then let $i = \mathrm{MinO}(f)$. Now, there must exist an x such that $f(x) \neq \varphi_{p(i)}(x)$. Thus, $\mathbf{M}$ on some initial segment of f, outputs $\perp$. Also, if $\mathbf{M}(\sigma) = \perp$, then $\mathbf{M}(\tau) = \perp$, for all extensions τ of σ.

It follows that $\mathbf{M}\ \mathbf{RefEx}^{a+1}$-identifies $\mathcal{C}$. ∎

Clearly, from Theorem 55 we immediately get the following hierarchy results, where the last one was already obtained in [22].

Corollary 56 *For every $a \in \mathbb{N}$,*

(1) $\mathbf{RefEx}^a \subset \mathbf{RefEx}^{a+1}$,
(2) $\mathbf{WRefEx}^a \subset \mathbf{WRefEx}^{a+1}$,
(3) $\mathbf{RelEx}^a \subset \mathbf{RelEx}^{a+1}$.

A proof similar to Lemma 54 can be used to show that

Lemma 57 *There exists a function $p \in \mathcal{R}$ such that, for all $i, j \in \mathbb{N}$:*

(a) $\mathrm{range}(\varphi_{p(\langle i,j\rangle)}) \subseteq \{0,1\}$.

(b) $\varphi_{p(\langle i,j\rangle)}$ *is undefined on at most $j+1$ inputs.*

(c) $\varphi_{p(\langle i,j\rangle)}(\langle i,j\rangle) = 1$, *and, for all $x < \langle i,j\rangle$, $\varphi_{p(\langle i,j\rangle)}(x) = 0$.*

(d) $\{f \in \mathcal{R}_{0,1} \mid \varphi_{p(\langle i,j\rangle)} \subseteq f\} \not\subseteq \mathbf{Ex}^j(\mathbf{M}_i)$.

Now a proof similar to the proof of Theorem 55 can be used to show the following result. Notice that $\mathbf{Ex}^* \setminus \bigcup_{a\in\mathbb{N}} \mathbf{Ex}^a \neq \emptyset$ was proved in [10].

Theorem 58 $\mathbf{RefEx}^* \setminus \bigcup_{a\in\mathbb{N}} \mathbf{Ex}^a \neq \emptyset$.

From Theorem 55 we can derive further corollaries. Therefore, we need the following notation. For $\eta \in \mathcal{P}$, let cyl_η be defined as follows: $\mathrm{cyl}_\eta(\langle x,y\rangle) = \eta(x)$. For $\mathcal{C} \subseteq \mathcal{R}$, let $\mathrm{cyl}_{\mathcal{C}} = \{\mathrm{cyl}_f | f \in \mathcal{C}\}$.

Proposition 59 (a) *If $\mathcal{C} \in \mathbf{RefBc}$, then $\mathrm{cyl}_{\mathcal{C}} \in \mathbf{RefBc}$.*

(b) $\mathrm{cyl}_{\mathcal{C}} \in \mathbf{Ex}^*$ *iff* $\mathrm{cyl}_{\mathcal{C}} \in \mathbf{Ex}$ *iff* $\mathcal{C} \in \mathbf{Ex}$.

Proof. (a) Suppose $\mathbf{M}\ \mathbf{RefBc}$-identifies $\mathcal{C}$.

For any σ, let uncyl_σ be defined as follows. Let $f_\sigma(x) = \sigma(\langle x,0\rangle)$. Let m be the smallest value such that f_σ is not defined. Then, let $\mathrm{uncyl}_\sigma = f_\sigma[m]$. It is easy to verify that, for all g, n, $\mathrm{uncyl}_{\mathrm{cyl}_g[n]} \subseteq g$. Moreover, $\lim_{n\to\infty} |\mathrm{uncyl}_{\mathrm{cyl}_g[n]}| = \infty$.

Let $\mathrm{progcyl}(p)$ be a program obtained effectively from p for cyl_{φ_p}. Now define

$\mathbf{M}'$ as follows:

$$\mathbf{M}'(\sigma) = \begin{cases} \perp, & \text{if } (\exists x, y, z)[\sigma(\langle x, y\rangle)\downarrow \neq \sigma(\langle x, z\rangle)\downarrow]; \\ \perp, & \text{if } \mathbf{M}(\text{uncyl}_\sigma) = \perp; \\ \text{progcyl}(p), & \text{if } \mathbf{M}(\text{uncyl}_\sigma) = p \text{ and} \\ & \neg(\exists x, y, z)[\sigma(\langle x, y\rangle)\downarrow \neq \sigma(\langle x, z\rangle)\downarrow]. \end{cases}$$

It is easy to verify that $\mathbf{M}'$ $\mathbf{RefBc}$-identifies $\text{cyl}_{\mathcal{C}}$.

Assertion (b) is an immediate consequence of the corresponding definitions. ∎

In [10], $\mathbf{Ex}^* \subseteq \mathbf{Bc}$ was shown. This result also holds for all of our types of refutable learning.

Proposition 60 *For* $\mathbf{I} \in \{\mathbf{Ref}, \mathbf{WRef}, \mathbf{Rel}\}$, $\mathbf{IEx}^* \subseteq \mathbf{IBc}$.

Proof. Suppose $\mathcal{C} \in \mathbf{IEx}^*$ as witnessed by $\mathbf{M}$. Here for $\mathbf{RelEx}^*$-identification, we assume that $\mathbf{M}$ is doing the identification in the sense of Definition 7 (that is if $f \notin \mathbf{Ex}^*(\mathbf{M})$, then $\mathbf{M}$ on f outputs $\perp$ infinitely often). Suppose $\mathcal{C} \in \mathbf{Bc}$ as witnessed by $\mathbf{M}'$ (since $\mathbf{Ex}^* \subseteq \mathbf{Bc}$, such an $\mathbf{M}'$ exists). Define $\mathbf{M}''$ as follows.

$$\mathbf{M}''(f[n]) = \begin{cases} \mathbf{M}'(f[n]), & \text{if } \mathbf{M}(f[n]) \neq \perp; \\ \perp, & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}''$ $\mathbf{IBc}$-identifies $\mathcal{C}$. ∎

In [10] it was proved that $\mathbf{Bc} \setminus \mathbf{Ex}^* \neq \emptyset$. This result holds refutably, as our next corollary shows.

Corollary 61 $\mathbf{RefBc} \setminus \mathbf{Ex}^* \neq \emptyset$.

Proof. By Theorem 55, there exists a class $\mathcal{C} \in \mathbf{RefEx}^1 \setminus \mathbf{Ex}$. Thus, by Proposition 60, there exists a class $\mathcal{C} \in \mathbf{RefBc} \setminus \mathbf{Ex}$. Now, $\text{cyl}_{\mathcal{C}} \in \mathbf{RefBc} \setminus \mathbf{Ex}^*$, by Proposition 59. ∎

The next corollary points out that already $\mathbf{RefEx}^1$ contains "algorithmically rich" classes of *predicates*.

Corollary 62 $\mathbf{RefEx}^1 \cap 2^{\mathcal{R}_{0,1}} \not\subseteq \mathbf{NUM} \cap 2^{\mathcal{R}_{0,1}}$.

Proof. Let $a = 0$, and let $\mathcal{C} \in 2^{\mathcal{R}_{0,1}}$ be defined as in the proof of Theorem 55. Then, by that proof, $\mathcal{C} \in \mathbf{RefEx}^1 \setminus \mathbf{Ex}$. Hence $\mathcal{C} \notin \mathbf{NUM}$, since $\mathbf{NUM} \subseteq \mathbf{Ex}$, see [15]. ∎

Corollary 62 can be even strengthened by replacing $\mathbf{RefEx}^1$ with **RefEx**. This another time exhibits the richness of already the most stringent of our types of learning refutably.

Theorem 63 $\mathbf{RefEx} \cap 2^{\mathcal{R}_{0,1}} \not\subseteq \mathbf{NUM} \cap 2^{\mathcal{R}_{0,1}}$.

Proof. Let $(\forall^n x)$ denote for all but at most n of x. That is, $(\forall^n x)[P(x)]$, denotes $\text{card}(\{x \mid \neg P(x)\}) \leq n$. In [8] it was shown that there exist recursive functions g and p such that for each e, the following three conditions are satisfied.

(a) $\varphi_{p(e)}(e) = 1$ and, for $x < e$, $\varphi_{p(e)}(x) = 0$.
(b) $\text{domain}(\varphi_{p(e)})$ is either $\mathbb{N}$ or an initial segment of $\mathbb{N}$ and $\text{range}(\varphi_{p(e)}) \subseteq \{0, 1\}$.
(c) If φ_e is total, then
  (c.1) $\varphi_{p(e)}$ is total,
  (c.2) $(\forall j)(\forall^{j+1}\, x > \max(\{e, j\}))[(\exists y \leq x)[\varphi_j(y) \neq \varphi_{p(e)}(y)] \vee [\Phi_{p(e)}(x) \leq g(e, x, \Phi_j(x))]]$, and
  (c.3) $(\forall j \mid \varphi_j = \varphi_{p(e)})(\forall^\infty x)[\Phi_j(x) > \varphi_e(x)]$.

Let $\mathcal{C} = \{\varphi_{p(e)} \mid \varphi_e \text{ is total}\}$. It was shown in [8] that this class contains arbitrarily complex functions from $\mathcal{R}_{0,1}$ (based on clause c.3 above), and thus $\mathcal{C} \notin \mathbf{NUM}$.

We now define an IIM that **RefEx**-learns $\mathcal{C}$. Let q be a program for Zero.

$$\mathbf{M}(f[n]) = \begin{cases} q, & \text{if } (\forall x < n)[f(x) = 0]; \\ p(e), & \text{if } e < n \;\&\; f(e) = 1 \;\&\; (\forall x < e)[f(x) = 0] \;\&\; \\ & \neg[(\exists x < n)[\Phi_{p(e)}(x) < n \;\&\; \varphi_{p(e)}(x) \neq f(x)]] \;\&\; \\ & \neg[(\exists j \leq n)(\exists S \subseteq \mathbb{N} \mid \text{card}(S) = j + 2, \min(S) > \max(\{e, j\})) \\ & [(\forall y \leq \max(S))[\Phi_j(y) \leq n \;\&\; \varphi_j(y) = f(y)] \;\&\; \\ & \quad (\forall x \in S)[\Phi_{p(e)}(x) > g(e, x, \Phi_j(x))]]]; \\ \perp, & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}$ $\mathbf{RefEx}$-identifies $\mathcal{C}$. Theorem follows. ∎

Note that Theorem 63 contrasts a known result on reliable **Ex**-learning. Actually, if we require the **Ex**-learning machine's reliability not only on the set $\mathcal{R}$ of all recursive functions, but even on the set of all *total* functions, then all the classes of recursive *predicates* belonging to this latter type turn out to be in **NUM**, see [16].

We now prove the analogue to Theorem 55 for $\mathbf{Bc}^a$-learning rather than $\mathbf{Ex}^a$-learning. Note that $\mathbf{Bc}^{a+1} \setminus \mathbf{Bc}^a \neq \emptyset$ was shown in [10]. We need the following lemma.

Lemma 64 *For any $a \in \mathbb{N}$, there exist a recursive function p and a partial recursive function q such that, for all i:*

(a) *The following three conditions hold:*

(a.1) *for all j,* $\text{range}(\varphi_{p(i,j)}) \subseteq \{0,1\}$,

(a.2) $\varphi_{p(i,0)}(i) = 1$,

(a.3) *for all $x < i$,* $\varphi_{p(i,0)}(x) = 0$.

(b) *Either $\{x \mid q(i,x)\downarrow\} = \mathbb{N}$ or there exists an s such that $\{x \mid q(i,x)\downarrow\} = \{x \mid x \leq s\}$.*

(c) *If $\{x \mid q(i,x)\downarrow\} = \mathbb{N}$, then the following two conditions hold:*

(c.1) *for all j,* $\varphi_{p(i,j)} =^{a+1} \varphi_{p(i,0)}$,

(c.2) $\varphi_{p(i,0)} \in \mathcal{R}$ *and* $\varphi_{p(i,0)} \notin \mathbf{Bc}^a(\mathbf{M}_i)$.

(d) *If $\{x \mid q(i,x)\downarrow\} = \{x \mid x \leq s\}$, then the following three conditions hold:*

(d.1) $\varphi_{p(i,0)} \subseteq \varphi_{p(i,s)}$,

(d.2) $\varphi_{p(i,s)} \in \mathcal{R}$ *and* $\varphi_{p(i,s)} \notin \mathbf{Bc}^a(\mathbf{M}_i)$,

(d.3) *for all j such that $1 \leq j < s$,* $\text{domain}(\varphi_{p(i,j)}) = \text{domain}(\varphi_{p(i,0)})$ *and* $\varphi_{p(i,j)} =^{a+1} \varphi_{p(i,0)}$.

Proof. The lemma is proved using a modification of the proof of $\mathbf{Bc}^{a+1} \setminus \mathbf{Bc}^a \neq \emptyset$ in [10]. By the Operator Recursion Theorem [7], there exists a recursive p such that $\varphi_{p(i,j)}$ may be defined as follows. For a fixed i, we will define $\varphi_{p(i,\cdot)}$, and $q(i,\cdot)$, in stages as follows. The construction can be easily seen to be effective in i. Note that if $\varphi_{p(i,j)}(x)$ (respectively $q(i,y)$) is not defined in stages below then $\varphi_{p(i,j)}(x)\uparrow$ (respectively, $q(i,y)\uparrow$). Initially, let

$q(i,0) = 0$,

$\varphi_{p(i,0)}(i) = 1$, and

$\varphi_{p(i,0)}(x) = 0$, for $x < i$.

Let x_s denote the least x such that $\varphi_{p(i,0)}(x)$ has not been defined before stage s. Thus, $x_1 = i + 1$. Go to stage 1.

Stage s
1. Let $q(i,s) = 0$.

For $x < x_s$, let $\varphi_{p(i,s)}(x) = \varphi_{p(i,0)}(x)$.

2. Let $f = 0\text{-ext}(\varphi_{p(i,0)})$.
3. Dovetail steps 4 and 5 until step 4 succeeds. If and when step 4 succeeds, go to step 6.
4. Search for a set S_s of cardinality $a+1$ and $m_s > x_s$ such that
   $(\forall x \in S_s)[x > m_s \ \& \ \varphi_{\mathbf{M}(f[m_s])}(x)\downarrow]$.
5. For $x = x_s$ to ∞ do
   Let $\varphi_{p(i,s)}(x) = 0$.
   EndFor
6. If and when such an S_s and m_s are found,
   let $w = \max(S_s \cup \{x \mid \varphi_{p(i,s)}(x)$ was defined in step 5 above $\})$.

6.1 For $x \in S_s$, let $\varphi_{p(i,0)}(x) = 1 \dot{-} \varphi_{\mathbf{M}(f[m_s])}(x)$.

6.2 For $x_s \leq x \leq w$ such that $x \notin S_s$, let $\varphi_{p(i,0)}(x) = 0$.

6.3 For $x_s \leq x \leq w$ such that $\varphi_{p(i,s)}(x)$ has not been defined upto now, let $\varphi_{p(i,s)}(x) = 0$.

6.4 Let $\varphi_{p(i,s)}$ follow $\varphi_{p(i,0)}$ from now on. That is, for $x > w$ such that $\varphi_{p(i,s)}(x)$ has not been defined upto now, $\varphi_{p(i,s)}(x)$ is made to be same as $\varphi_{p(i,0)}(x)$ whenever, if ever, $\varphi_{p(i,0)}(x)$ gets defined.
(* This ensures that $\varphi_{p(i,s)}$ and $\varphi_{p(i,0)}$ are same on all $x \notin S_s$ *).

6.5 Go to stage $s+1$.
(* Note that $x_{s+1} = w+1$. *)

End stage s

Fix i. Clearly, parts (a) and (b) of Lemma are satisfied. To show parts (c) and (d) we consider two cases.

Case 1. All stages terminate.

In this case $q(i,x)$ is defined for all x. (c.1) holds due to step 6.4. (Note that $\varphi_{p(i,s)}$ and $\varphi_{p(i,0)}$ are same on all $x \notin S_s$).

Also, due to step 6.1, for all s, $\varphi_{\mathbf{M}(\varphi_{p(i,0)}[m_s])} \neq^a \varphi_{p(i,0)}$. Thus (c.2) is satisfied.

Case 2. Stage s starts but does not terminate.

In this case $q(i,x)$ is defined for $x \leq s$, and undefined for $x > s$. (d.1) clearly holds due to step 1. (d.2) holds since, for all but finitely many m, $\varphi_{\mathbf{M}(\varphi_{p(i,s)}[m])}$ is finite (otherwise step 4 would succeed). Comment at the end of step 6.4 implies (d.3).

This proves the lemma. ∎

Theorem 65 *For all $a \in \mathbb{N}$, $\mathbf{RefBc}^{a+1} \setminus \mathbf{Bc}^a \neq \emptyset$.*

Proof. Let p and q be as in Lemma 64. Let

$$\mathcal{C}_i = \begin{cases} \{\varphi_{p(i,0)}\}, & \text{if } \{x \mid q(i,x)\downarrow\} = \mathbb{N}. \\ \{f \in \mathcal{R}_{0,1} \mid \varphi_{p(i,0)} \subseteq f \ \&\ f =^{a+1} \varphi_{p(i,s)}\}, & \text{if } \{x \mid q(i,x)\downarrow\} = \{x \mid x \leq s\}. \end{cases}$$

Let $\mathcal{C} = \{\text{Zero}\} \cup \bigcup_{i \in \mathbb{N}} \mathcal{C}_i$.

We claim that $\mathcal{C} \in \mathbf{RefBc}^{a+1} \setminus \mathbf{Bc}^a$. By Lemma 64, it follows that $\mathcal{C}_i \not\subseteq \mathbf{Bc}^a(\mathbf{M}_i)$. Thus, $\mathcal{C} \notin \mathbf{Bc}^a$.

Let

$$\text{MinO}(\sigma) = \min(\{x \mid \sigma(x) = 1\}),$$

$$\begin{aligned} Z_1 = &\{\sigma \in \text{SEG}_{0,1} \mid \sigma \not\subseteq \text{Zero} \ \&\ \text{MinO}(\sigma) = i \ \ \&\ \ (\exists x \in \mathbb{N})[\varphi_{p(i,0)}(x)\downarrow \neq \sigma(x)\downarrow]\}, \\ Z_2 = &\{\sigma \in \text{SEG}_{0,1} \mid \sigma \not\subseteq \text{Zero} \ \&\ \text{MinO}(\sigma) = i \ \ \& \\ &\quad (\exists s \in \mathbb{N})(\exists S \subseteq \mathbb{N} \mid \text{card}(S) = a+2)[q(i,s)\downarrow \ \&\ (\forall x \in S)[\varphi_{p(i,s)}(x)\downarrow \neq \sigma(x)\downarrow]]\}. \end{aligned}$$

The following claim follows easily from the definition of $\mathcal{C}$.

Claim 66 *$f \in \mathcal{C}$ iff $(\forall n \in \mathbb{N})[f[n] \notin Z_1 \cup Z_2]$.*

Note that Z_1 and Z_2 are recursively enumerable. Let Z_1^s and Z_2^s respectively denote Z_1, Z_2 enumerated upto s steps in some standard recursive enumeration.

Now define $\mathbf{M}$ as follows. Let z denote a program for Zero.

$$\mathbf{M}(\sigma) = \begin{cases} z, & \text{if } \sigma \subseteq \text{Zero}; \\ \perp, & \text{if } (\exists \tau \subseteq \sigma)[\tau \in Z_1^{|\sigma|} \cup Z_2^{|\sigma|}]; \\ p(i,s), & \text{otherwise, where } \text{MinO}(\sigma) = i, \text{ and} \\ & s = \max(\{x \leq |\sigma| \mid q(i,x) \text{ converges within } |\sigma| \text{ steps}\}). \end{cases}$$

It is easy to verify that,

(i) if $f \notin \mathcal{C}$, then $\mathbf{M}(f[m]) = \perp$ for all but finitely many m. Moreover, $\mathbf{M}(\sigma) = \perp$ implies $\mathbf{M}(\sigma') = \perp$ for all extensions σ' of σ,

(ii) if $f = \text{Zero}$ then $\mathbf{M}$ outputs z as its only program on f,

(iii) if $f \in \mathcal{C}_i$ and $\{x \mid q(i,x)\downarrow\} = \mathbb{N}$, then $\mathbf{M}$ $\mathbf{Bc}^{a+1}$-identifies f, due to property (c.1) in Lemma 64,

(iv) if $f \in \mathcal{C}_i$ and $\{x \mid q(i,x)\downarrow\} = \{x \mid x \leq s\}$, then $\mathbf{M}(f)$ converges to $p(i,s)$, which is an $a+1$ error program for f (by definition of $\mathcal{C}_i$).

It thus follows that $\mathcal{C} \in \mathbf{RefBc}^{a+1}$. ∎

Again, Theorem 65 yields the following hierarchies, where the last one solves an open problem from [23].

Corollary 67 *For every $a \in \mathbb{N}$,*

(1) $\mathbf{RefBc}^a \subset \mathbf{RefBc}^{a+1}$,
(2) $\mathbf{WRefBc}^a \subset \mathbf{WRefBc}^{a+1}$,
(3) $\mathbf{RelBc}^a \subset \mathbf{RelBc}^{a+1}$.

Proposition 68 $\mathbf{RefBc}^* \setminus \bigcup_{a\in\mathbb{N}} \mathbf{Bc}^a \neq \emptyset$.

Proof. Since $\mathcal{R} \in \mathbf{Bc}^*$, see [10], we have that $\mathcal{R} \in \mathbf{RefBc}^*$. Since $\mathcal{R} \notin \bigcup_{a\in\mathbb{N}} \mathbf{Bc}^a$, see [10], proposition follows. ∎

Recall that in the proof of Theorem 16 we have derived that *FINSUP* $\notin$ **WRefEx**. This result will now be strengthened for $\mathbf{WRefBc}^a$-learning and then used in the next corollary below.

Theorem 69 *For every $a \in \mathbb{N}$, FINSUP $\notin \mathbf{WRefBc}^a$.*

Proof. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{WRefBc}^a$-identifies *FINSUP*.

Claim 70 (a) *For all σ, there exists a $\tau \supseteq \sigma$ such that $\mathbf{M}(\tau) \neq \perp$.*
(b) *For all σ, there exists a $\tau \supseteq \sigma$ such that $\mathbf{M}(\tau) = \perp$.*

Proof. Part (a) holds, since otherwise $\mathbf{M}$ does not $\mathbf{Bc}$-identify 0-ext(σ). Part (b) holds since $\mathbf{M}$ cannot $\mathbf{Bc}^a$-identify all extensions of σ. □

Now define σ_i, τ_i as follows. $\sigma_0 = \Lambda$. Let τ_i be an extension of σ_i such that $\mathbf{M}(\tau_i) = \perp$. Let σ_{i+1} be an extension of τ_i such that $\mathbf{M}(\tau_i) \neq \perp$. Note that all σ_i and τ_i are defined by Claim 70 and can be effectively obtained. Now $\mathbf{M}$ outputs $\perp$ infinitely often on $\bigcup_{i\in\mathbb{N}} \sigma_i$, without converging to $\perp$. A contradiction to $\mathbf{M}$ $\mathbf{WRefBc}^a$-identifying *FINSUP*. Theorem follows. ∎

The following corollary points out the relative strength of **RelEx**-learning over $\mathbf{WRefBc}^a$-learning. In other words, in general, one cannot compensate a stricter refutability constraint by a more liberal learning criterion.

Corollary 71 *For all $a \in \mathbb{N}$, $\mathbf{RelEx} \setminus \mathbf{WRefBc}^a \neq \emptyset$.*

Proof. Follows from Theorem 69, since $FINSUP \in \mathbf{RelEx}$. ∎

Our final result exhibits the strength of **WRefEx**-learning over $\mathbf{RefBc}^a$-learning. Thus, it is in the same spirit as Corollary 71 above.

Theorem 72 *For all $a \in \mathbb{N}$, $\mathbf{WRefEx} \setminus \mathbf{RefBc}^a \neq \emptyset$.*

Proof. Let

$$\sigma_i(x) = \begin{cases} 0, & \text{if } x < i; \\ 1, & \text{if } x = i; \\ \uparrow, & \text{otherwise.} \end{cases}$$

Define $\mathcal{C}_i$ as follows:

$$\mathcal{C}_i = \begin{cases} \{0\text{-ext}(\tau)\}, & \text{if } \tau \text{ is the least extension (in } \mathrm{SEG}_{0,1}) \text{ of } \sigma_i \\ & \text{such that } \mathbf{M}_i(\tau) = \perp; \\ \emptyset, & \text{otherwise.} \end{cases}$$

Let $\mathcal{C} = \{\text{Zero}\} \cup \bigcup_{i \in \mathbb{N}} \mathcal{C}_i$. It is easy to verify that $\mathcal{C} \in \mathbf{WRefEx}$. Suppose by way of contradiction that $\mathbf{M}_i$ $\mathbf{RefBc}^a$-identifies $\mathcal{C}$. Then, we consider two cases.

Case 1. There exists an extension τ' (in $\mathrm{SEG}_{0,1}$) of σ_i such that $\mathbf{M}_i(\tau') = \perp$.

In this case, let τ be least such τ'. Now $0\text{-ext}(\tau) \in \mathcal{C}_i$, but $\mathbf{M}_i(\tau) = \perp$.

Case 2. There does not exist an extension τ' (in $\mathrm{SEG}_{0,1}$) of σ_i such that $\mathbf{M}_i(\tau') = \perp$.

In this case $\mathbf{M}_i$ must $\mathbf{Bc}^a$-identify all $f \in \mathcal{R}_{0,1}$ such that $\sigma_i \subseteq f$. However, this contradicts Proposition 51.

From the above cases it follows that $\mathbf{M}_i$ does not $\mathbf{RefBc}^a$-identify $\mathcal{C}$. ∎

Note that Theorems 53, 69 and 72, and Corollary 71 hold even if we replace $\mathbf{Bc}^a$ by any criterion of inference for which Proposition 51 holds.

Acknowledgement

We would like to thank the referees for their valuable comments and suggestions which have resulted in several improvements of the presentation of the paper.

References

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

[2] 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.

[3] J. Bārzdiņš and R. Freivalds. Prediction and limiting synthesis of recursively enumerable classes of functions. *Latvijas Valsts Univ. Zimatm. Raksti*, 210:101–111, 1974.

[4] S. Ben-David. Can finite samples detect singularities of real-valued functions? In *Symposium on the Theory of Computation*, pages 390–399, 1992.

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

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

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

[8] J. Case, S. Jain, and S. Ngo Manguelle. Refinements of inductive inference by Popperian and reliable machines. *Kybernetika*, 30:23–52, 1994.

[9] J. Case, E. Kinber, A. Sharma, and F. Stephan. On the classification of computable languages. In R. Reischuk and M. Morvan, editors, *Proc. 14th Symposium on Theoretical Aspects of Computer Science*, volume 1200 of *Lecture Notes in Computer Science*, pages 225–236. Springer-Verlag, 1997.

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

[11] R. Freivalds. Inductive inference of recursive functions: Qualitative theory. In J. Bārzdiņš and D. Bjorner, editors, *Baltic Computer Science*, volume 502 of *Lecture Notes in Computer Science*, pages 77–110. Springer-Verlag, 1991.

[12] R. Freivalds, J. Bārzdiņš, and K. Podnieks. Inductive inference of recursive functions: Complexity bounds. In J. Bārzdiņš and D. Bjørner, editors, *Baltic Computer Science*, volume 502 of *Lecture Notes in Computer Science*, pages 111–155. Springer-Verlag, 1991.

[13] R. Freivalds, E. Kinber, and C. Smith. On the intrinsic complexity of learning. *Information and Computation*, 123(1):64–71, 1995.

[14] E. M. Gold. Limiting recursion. *Journal of Symbolic Logic*, 30:28–48, 1965.

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

[16] J. Grabowski. Starke Erkennung. In R. Lindner and H. Thiele, editors, *Strukturerkennung diskreter kybernetischer Systeme, Teil I*, pages 168–184. Seminarbericht Nr.82, Department of Mathematics, Humboldt University of Berlin, 1986.

[17] G. Grieser. Reflecting inductive inference machines and its improvement by therapy. In S. Arikawa and A. Sharma, editors, *Algorithmic Learning Theory: Seventh International Workshop (ALT '96)*, volume 1160 of *Lecture Notes in Artificial Intelligence*, pages 325–336. Springer-Verlag, 1996.

[18] S. Jain. Learning with refutation. *Journal of Computer and System Sciences*, 57(3):356–365, 1998.

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

[20] K. P. Jantke. Reflecting and self-confident inductive inference machines. 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 282–297. Springer-Verlag, 1995.

[21] W. Jekeli. *Universelle Strategien zur Lösung induktiver Lernprobleme*. PhD thesis, Dept. of Computer Science, University of Kaiserslautern, 1997. MSc Thesis.

[22] E. Kinber and T. Zeugmann. Inductive inference of almost everywhere correct programs by reliably working strategies. *Journal of Information Processing and Cybernetics (EIK)*, 21:91–100, 1985.

[23] E. Kinber and T. Zeugmann. One-sided error probabilistic inductive inference and reliable frequency identification. *Information and Computation*, 92:253–284, 1991.

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

[25] S. Lange and P. Watson. Machine discovery in the presence of incomplete or ambiguous data. In S. Arikawa and K. Jantke, editors, *Algorithmic Learning Theory: Fourth International Workshop on Analogical and Inductive Inference (AII '94) and Fifth International Workshop on Algorithmic Learning Theory (ALT '94)*, volume 872 of *Lecture Notes in Artificial Intelligence*, pages 438–452. Springer-Verlag, 1994.

[26] R. Lindner. *Algorithmische Erkennung*. PhD thesis, University of Jena, 1972. In German.

[27] M. Machtey and P. Young. *An Introduction to the General Theory of Algorithms*. North Holland, New York, 1978.

[28] E. Minicozzi. Some natural properties of strong identification in inductive inference. *Theoretical Computer Science*, 2:345–360, 1976.

[29] T. Miyahara. Refutable inference of functions computable by loop programs. Technical Report RIFIS-TR-CS-112, Kyushu University, Fukuoka, 1995.

[30] Y. Mukouchi and S. Arikawa. Inductive inference machines that can refute hypothesis spaces. In K.P. Jantke, S. Kobayashi, E. Tomita, and T. Yokomori, editors, *Algorithmic Learning Theory: Fourth International Workshop (ALT '93)*, volume 744 of *Lecture Notes in Artificial Intelligence*, pages 123–136. Springer-Verlag, 1993.

[31] Y. Mukouchi and S. Arikawa. Towards a mathematical theory of machine discovery from facts. *Theoretical Computer Science*, 137:53–84, 1995.

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

[33] K. R. Popper. *The Logic of Scientific Discovery*. Harper and Row, 1965.

[34] H. Rice. On completely recursively enumerable classes and their key arrays. *Journal of Symbolic Logic*, 21:304–308, 1956.

[35] H. Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, 1967. Reprinted by MIT Press in 1987.

[36] C. Smith, R. Wiehagen, and T. Zeugmann. Classifying predicates and languages. *International Journal of Foundations of Computer Science*, 8:15–41, 1997.

[37] F. Stephan. On one-sided versus two-sided classification. Technical Report Forschungsberichte Mathematische Logik 25/1996, Mathematical Institute, University of Heidelberg, 1996.

[38] R. Wiehagen. Characterization problems in the theory of inductive inference. In G. Ausiello and C. Böhm, editors, *Proceedings of the 5th International Colloquium on Automata, Languages and Programming*, volume 62 of *Lecture Notes in Computer Science*, pages 494–508. Springer-Verlag, 1978.

[39] R. Wiehagen and C. H. Smith. Generalization versus classification. *Journal of Experimental and Theoretical Artificial Intelligence*, 7:163–174, 1995.

[40] T. Zeugmann. A-posteriori characterizations in inductive inference of recursive functions. *Journal of Information Processing and Cybernetics (EIK)*, 19:559–594, 1983.

[41] T. Zeugmann. *Algorithmisches Lernen von Funktionen und Sprachen*. Habilitationsschrift, Technical University of Darmstadt, 1993.