2-1997

On the Classification of Computable Languages

John Case
University of Delaware, case@cis.udel.edu

Efim Kinber
Sacred Heart University, kinbere@sacredheart.edu

Arun Sharma
University of New South Wales

Frank Stephan
University of Heidelberg

On the Classification of Computable Languages

John Case * University of Delaware
Efim Kinber † Sacred Heart University
Arun Sharma ‡ University of New South Wales
Frank Stephan § University of Heidelberg

Abstract

A one-sided classifier converges to 1 on every set inside a given class and outputs infinitely often a 0 on every set outside the class. A two-sided classifier converges in the first case to 1 and in the second to 0. This paper considers one-sided and two-sided classifiers dealing with computable sets as input. It provides theorems from which the classifiability of natural examples can be assessed and investigates the relations of the types of classification to inductive learning theory and structural complexity theory in terms of Turing degrees. Furthermore, it deals with the special cases of classification from positive data only and of inferring trial-and-error classifier programs.

1 Introduction

Consider the problem of determining whether a language A over $\mathbb{N}$, the set of natural numbers $\{0, 1, 2, \ldots\}$, satisfies a certain property. Let $\mathcal{A}$ denote the class of all languages over $\mathbb{N}$ that satisfy the given property. The question of classification then is: if one is given data about A, can one determine if A is a member of $\mathcal{A}$.

We briefly discuss the various approaches to the study of classification in the literature. One of the earliest attempts was the design of finite automata to decide whether an infinite string (representing the characteristic function of a language) belongs to a given ω-language or not [8, 21, 23, 32]. But the restrictive computational ability of these finite automata led Büchi [8] and his successors to consider non-deterministic automata. The present paper takes the alternate approach of choosing Turing machines as classifiers. In fact this approach had already been begun by Büchi and Landweber [9, 20].

Smith and Wiehagen [30] introduced a model of classification analogous to the Gold model of learning [16]. The (computable) classifier M sees longer and longer prefixes σ of the characteristic function of a language $A \in \mathcal{A}_1 \cup \mathcal{A}_2 \cup \ldots \cup \mathcal{A}_k$ and guesses on each input σ some number $h \in \{1, 2, \ldots, k\}$ to indicate that $A \in \mathcal{A}_h$. These guesses are supposed to converge, for each set $A \in \mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$, to a value h such that $A \in \mathcal{A}_h$. Smith, Wiehagen and Zeugmann [31] extended this study in various ways.

Ben-David [5] and Kelly [18] also interestingly studied classification. They call a class *classifiable* iff there exists a (not-necessarily-computable) functional that indicates in the limit for every A whether or not it belongs to a given class $\mathcal{A}$. They obtained topological conditions for classifiable classes. Gasarch, Pleszkoch, Stephan and Velauthapillai [14] extended this study and obtained relations between the Borel hierarchy on classes – which is induced by the space $\{0, 1\}^\infty$ with product topology – and the query hierarchy obtained by allowing a certain number of quantifier-alternations during querying a teacher on the target set A.

Later Stephan [28] investigated the limits of (computable) classifiers. He considered classification of languages w.r.t. one single class $\mathcal{A}$ and introduced two models of classification. Our study derives from these

*Computer and Information Sciences Department, 101A Smith Hall, University of Delaware, Newark, DE 19716-2586, U.S.A., case@cis.udel.edu.

†Computer Science, Sacred Heart University, 5151 Park Avenue, Fairfield, CT 06432-1000, U.S.A., kinber@shu.sacredheart.edu.

‡School of Computer Science and Engineering, The University of New South Wales, Sydney, NSW, 2052, Australia, arun@cse.unsw.edu.au. Supported by a grant from the Australian Research Council.

§Mathematisches Institut der Universität Heidelberg, Im Neuenheimer Feld 294, 69120 Heidelberg, Germany, fstephan@math.uni-heidelberg.de. Supported by the Deutsche Forschungsgemeinschaft (DFG) grant Am 60/9-1.

models which we present next. But, first some notation.

We take a *classifier* to be an algorithmic device; M, N and H ranges over classifiers. Calligraphic letters range over classes, A, B over sets and U over oracles. We take σ, τ range over prefixes of strings or characteristic functions of sets. $\sigma \preceq \tau$ means that $\tau(x)\downarrow = \sigma(x)$ for all $x \in dom(\sigma)$. $M(\sigma)$ denotes the guess issued by classifier M on a prefix $\sigma \preceq A$ of the input-set A.

Two-Sided Classification: For all languages A: $M(\sigma) = \mathcal{A}(A)$ for almost all $\sigma \preceq A$.

Here $\mathcal{A}(A)$ is 1 if $A \in \mathcal{A}$ and 0 otherwise, i.e., classes and sets are identified with their characteristic function. Two-sided classification may be considered to be a too strong requirement. In some applications it is sufficient if the classifier is able to signal the inclusion of a language in a given class, but only provides a weaker signal if the language is not in the class. Stephan [28] introduced the notion of one-sided classification to model this idea.

One-Sided Classification: For all languages A: if $A \in \mathcal{A}$, then $M(\sigma) = 1$ for almost all $\sigma \preceq A$; if $A \notin \mathcal{A}$, then $M(\sigma) = 0$ for infinitely many $\sigma \preceq A$.

We normally let M and N range over two-sided classifiers and H range over one-sided classifiers. The notion of one-sided classification is reasonable since the classifier outputs 0 infinitely often thereby guaranteeing that the classifier never locks onto an incorrect conjecture.

In the present paper, we restrict our investigation to classification of computable languages. This restriction may be supported by the fact that practical examples are always computable, and assuming an algorithmic view of the universe, it is unlikely that nature generates noncomputable languages. Thus, our classifiers can be relied upon if they are never expected to deliberate upon noncomputable languages. Hence, in the sequel, the statement "for all languages A" in the above two definitions is replaced by "for all computable languages A".[1]

The present paper may also be seen as closing the gap between Stephan's abstract work [28] and the more concrete approach of Smith, Wiehagen and Zeugmann [30, 31]. Before we begin a formal presentation of the results, we give an informal tour of the various sections in the paper.

In Section 2, we introduce the basic definitions and give preliminary results about two-sided and one-sided classification for classes of computable languages. We give concrete classes of languages that can be two-sided and one-sided classified. In particular we observe that one-sided classes are closed under finite monotone Boolean combinations and two-sided classes are closed under all finite Boolean combinations. We also show that every uniformly recursive family of languages is one-sided classifiable. Additionally, if the family is discrete, then it is also two-sided classifiable. As a consequence of this result, the class of pattern languages is two-sided classifiable. As a contrast, however, the class of regular languages is only one-sided classifiable.

Although, from [30] we already know that learning and classification are, in general, incomparable, in Section 3, we provide some pleasant links between learning and classification. We show that for classes identifiable in the limit from informant that they can be *reliably* identified iff they are one-sided classifiable. We also investigate conditions under which reliable identification in the limit and two-sided classification are linked. We show that if a class can be reliably identified with a constructive ordinal bound on the mind changes, then it is two-sided classifiable. However, the converse of this result is not true.

The characteristic function of a language conveys both positive and negative data about the language. In Section 4, we argue that it may not be realistic to assume the availability of both positive and negative data in practice. The experience from empirical studies of learning is that negative data is not always readily available and even when it is available, it is often tedious to obtain. Motivated by such concerns, we also investigate two-sided and one-sided classification from only positive data. Following the practice in inductive inference literature, we model positive data as texts. As expected, we show that classification from texts is very difficult. As a simple consequence of our result, the class of pattern languages is not even one-sided classifiable from texts.

Not deterred by the difficulty of classification from texts, we find a weaker version of classification for text presentation, called *partial classification*, about which one there are positive results. A class $\mathcal{A}$ is *partially*

[1] So, we ignore noncomputable sets everywhere. Accordingly, set-theoretic notions like the complement of classes are adapted to the computable universe: $\overline{\mathcal{A}} = \{\text{computable } A : A \notin \mathcal{A}\}$.

classifiable just in case there exists a machine that on texts for languages in $\mathcal{A}$ outputs exactly one guess infinitely often and on texts for nonmembers of $\mathcal{A}$ does not output infinitely often any guess. The motivation here is that a partial classifier gives a weak signal if the language belongs to the class and refuses to give any signal if the language is not a member of the class being classified. We show that each uniformly recursive family of computable languages is partially classifiable. We also give a sufficient condition for partial classification from texts in terms of classification from both positive and negative data. We show that if a class is one-sided classifiable from both positive and negative data, then it is partially classifiable from texts. The converse, however, does not hold.

In Section 5, we investigate structurally the computational limits of classifying computable languages. In particular, we investigate the "computational distance" between one-sided and two-sided classification by determining the kind of noncomputable information that yields a two-sided classifier for a class that was otherwise only one-sided classifiable. This gives insight into what it takes for a class of interest to be two- *vs* one-sided classifiable. We show that access to a high oracle is sufficient to construct a two-sided classifier for a one-sided classifiable class. We also establish that in some cases the power of a high oracle is necessary as there are classes for which any two-sided classifier has high Turing degree. We adapt Post's notion of creative set to describe the one-sided classifiable classes that are, effectively not two-sided classifiable. We call a one-sided classifiable class $\mathcal{A}$ *creative* just in case there is a uniformly computable sequence of languages $A_0, A_1, \ldots$ such that for each one-sided classifier H_e, the language A_e is a counterexample to the hypothesis "H_e classifies $\overline{\mathcal{A}}$". The analog between the two notions of creative is seen to be quite striking. We give examples of creative classes and show that a creative class is two-sided *only* relative to a high oracle. We discuss some interesting results about one-sided classifiable classes of intermediate complexity and compare our results with the more abstract study of classification by Stephan [28] in which a classifier has to behave correctly on noncomputable languages, too.

Finally, in Section 6, we consider classifiers that, instead of guessing 0 or 1, output programs that converge in the limit to 0 or 1. Such programs may be viewed as generators of trial and error guesses, and classifiers that output such programs may be viewed to be of somewhat lower quality (compared to the classifiers that directly guess 0 or 1). We consider two kinds of such classifiers: Ex-style requiring that the sequence of programs converge to a single program that has the correct guess of 0 or 1 in the limit and BC-style requiring that the sequence of programs eventually contain only programs that have the correct guess of 0 or 1 in the limit. We show that the notion of Ex-style classification nicely coincides with two-sided classification. We also show that every one-sided classifier has a BC-style classifier. We conclude with insightful, structural characterizations of BC-style classification.

We now proceed formally.

2 Basic Definitions and Results

Definition 2.1 A classifier H is an algorithm which outputs for every string σ a number 0 or 1. It classifies a class $\mathcal{A}$ *one-sided* iff

- if $A \in \mathcal{A}$, then $H(\sigma) = 1$ for almost all $\sigma \preceq A$; and
- if $A \in \overline{\mathcal{A}}$, then $H(\sigma) = 0$ for infinitely many $\sigma \preceq A$.

The classifier H is furthermore *two-sided* iff the statement "for infinitely many" in the second clause can be strengthened to "for almost all". Note that in this definition the variable A always ranges over only *computable* sets.

There is an effective list of classifiers H_e such that for each one-sided class there is some H_e classifying it one-sided and for each two-sided class there is some H_e classifying it two-sided. Assuming an acceptable numbering φ_e of all partial computable functions, these classifiers are defined as follows:

$$H_e(\sigma) = \begin{cases} \varphi_e(\tau) & \text{for the longest } \tau \preceq \sigma \text{ such that } \varphi_e(\tau) \text{ outputs 0 or 1 within } |\sigma| \text{ steps;} \\ 0 & \text{if there is no such } \tau. \end{cases}$$

Now it is easy to verify that whenever φ_e is a one-sided classifier for $\mathcal{A}$, then so is H_e; and whenever φ_e is a two-sided classifier for $\mathcal{A}$, then so is H_e. This normalization has the advantage that now we can assume without loss of generality that all one-sided (two-sided) classes have a total and computable one-sided (two-sided) classifier. Therefore, in the sequel, we will consider H_e instead of the underlying φ_e.

One-sided classes are closed under finite monotone Boolean combinations and two-sided classes are closed under all finite Boolean combinations. These facts follow from the following theorem.

Theorem 2.2 *$\mathcal{A}$ is two-sided iff $\mathcal{A}$ and $\overline{\mathcal{A}}$ are one-sided. If $\mathcal{A}, \mathcal{B}$ are one-sided classes so are $\mathcal{A} \cup \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}$. If $\mathcal{A}$ is one-sided so is $\mathcal{B} = \{B : B$ is a finite variant of some $A \in \mathcal{A}\}$.*

Proof The direction ($\Rightarrow$) of the first statement is obvious. For the reverse direction ($\Leftarrow$), let H_1 be a one-sided classifier for $\mathcal{A}$ and let H_2 be one for $\overline{\mathcal{A}}$. Let $M(\lambda) = 0$; we define inductively:

$$M(\sigma w) = \begin{cases} H_1(\sigma w) & \text{if } H_1(\sigma w) \neq H_2(\sigma w); \\ M(\sigma) & \text{otherwise.} \end{cases}$$

We claim that M is a two-sided classifier for $\mathcal{A}$: If a computable set A is in $\mathcal{A}$, then H_1 converges on A to 1 while H_2 outputs on A infinitely many 0s. So there are infinitely many $\tau \preceq A$ with $H_1(\tau) = 1$ and $H_2(\tau) = 0$ but only finitely many $\tau \preceq A$ with $H_1(\tau) = 0$ and $H_2(\tau) = 1$. So M will converge to 1. Similarly M will converge to 0 on any computable set $A \in \overline{\mathcal{A}}$.

For the second statement, let H_1 be a one-sided classifier for $\mathcal{A}$ and H_2 be a one-sided classifier for $\mathcal{B}$. Now $\mathcal{A} \cap \mathcal{B}$ has the one-sided classifier

$$H(\sigma) = \begin{cases} 1 & \text{if } H_1(\sigma) = 1 \text{ and } H_2(\sigma) = 1; \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to see that H outputs on A infinitely many 0s iff either H_1 or H_2 does. So H converges on A to 1 iff both H_1 and H_2 converge on A to 1. The case $\mathcal{A} \cup \mathcal{B}$ is a bit more involved. The following fact is used in defining the machine:

"H outputs on A in total at least n 0s if H_1 and H_2 both output on A in total at least n 0s."

This informal idea can be turned into an algorithm as follows: let

$$n_M(\sigma) = |\{\tau \preceq \sigma : M(\tau) = 0\}|$$

for each machine $M \in \{H, H_1, H_2\}$, $H(\lambda) = 1$ and

$$H(\sigma w) = \begin{cases} 0 & \text{if } n_{H_1}(\sigma w) > n_H(\sigma) \text{ and } n_{H_2}(\sigma w) > n_H(\sigma); \\ 1 & \text{otherwise.} \end{cases}$$

It is easy to see that H outputs infinitely many 0s iff both H_1 and H_2 output infinitely many 0s. Therefore, H converges on a set A to 1 if at least one of the machines H_1 and H_2 converge to 1.

The classifier for $\mathcal{B}$ in the last statement is constructed such that it outputs on B at least n 0s iff the classifier for $\mathcal{A}$ outputs on each set of the form $a_0 a_1 \ldots a_n B(n+1)B(n+2)\ldots$ at least n 0s. ∎

Given a computable function $A(x, y)$, let $A_x = \{y : A(x, y) = 1\}$ and $\mathcal{A} = \{A_0, A_1, \ldots\}$. Such an $\mathcal{A}$ is called a *uniformly recursive family*. Angluin [2] initiated the study of learning uniformly recursive families from texts and after the introduction of monotonicity constraints many papers have considered the learnability of these families from texts and informants [17, 33, 34]. A class $\mathcal{A}$ is *closed* iff for each $A \notin \mathcal{A}$ there is a $\sigma \preceq A$ such that no $B \in \mathcal{A}$ extends σ.

Theorem 2.3 *Every uniformly computable family is one-sided. If it is also closed, then it is two-sided.*

Proof As in the proof of Theorem 2.2 let $n_H(\sigma) = |\{\tau \preceq \sigma : H(\tau) = 0\}|$, $H(\lambda) = 1$ and

$$H(\sigma w) = \begin{cases} 1 & \text{if } \sigma w \preceq A_x \text{ for } x = n_N(\sigma); \\ 0 & \text{otherwise.} \end{cases}$$

The intuitive idea behind H is to check the sets $A_0, A_1, \ldots$; whenever A_x turns out to be different from A, H outputs a 0 and moves on to A_{x+1}, otherwise H outputs 1 as long as A_x and A appear to be equal. So H converges on every set A_x to 1 making (at most) x 0s and outputs infinitely many 0s for all $A \notin \mathcal{A}$.

Assume now the same algorithm for a closed class $\mathcal{A}$ and let $A \notin \mathcal{A}$ be computable. Then there is $\tau \preceq A$ such that no A_x extends τ. In particular $\sigma w \not\preceq A_x$ for all $\sigma \succeq \tau$, all x and almost all $\sigma w \preceq A$. If follows that $H(\sigma w) = 0$ for almost all $\sigma w \preceq A$. So H is already a two-sided classifier for $\mathcal{A}$. ▮

Example 2.4 The immediately preceding results yield the following examples.

- $\mathcal{C} = \{A : A \text{ is cofinite}\}$ is one-sided, but not two-sided.
  The classifier is $H(\sigma w) = w$.

- $\mathcal{D} = \{1^\infty, 01^\infty, 001^\infty, 0001^\infty, \ldots\}$ is two-sided.
  The classifier M outputs 1 if $\sigma \in 0^*1^+$ and 0 otherwise.

- $\mathcal{E} = \{A : A \text{ has finite and even cardinality}\}$ is one-sided, but not two-sided.
  The classifier $H(\sigma)$ outputs 1 iff the number of 1s in σ is even and 0 iff this number is odd.

- $\mathcal{F}_\phi = \{A : \text{the formula } \phi(A) \text{ is true}\}$ is two-sided.
  Here $\phi(A)$ means that ϕ is a Boolean formula, such as $[5 \in A \vee [3 \notin A \wedge 4 \notin A]]$, with A being the only free variable representing the input-set A of the same name. Such formulas can be evaluated after having seen a sufficiently long part of the input and from then on the classifier outputs 1 if $\phi(A)$ holds and 0 if $\phi(A)$ does not hold.

- $\mathcal{G} = \{graph(p) : p \text{ is a polynomial}\}$ is one-sided, but not two-sided.
  $\mathcal{G}$ and $\mathcal{R}$ below are uniformly recursive families and, hence, have the one-sided classifier from Theorem 2.3.

- $\mathcal{P} = \{A : A \text{ is a pattern language}\}$ is two-sided.
  This is due to the fact that the class of the pattern languages is both closed and uniformly recursive.

- $\mathcal{R} = \{A : A \text{ is regular}\}$ is one-sided, but not two-sided.

3 Links Between Learning and Classification

Reliable identification in the limit [22] means that the learner either diverges or converges to a correct index, but it never converges to a false one. So, the inferred class is also in some sense classified since convergence indicates membership in the class and divergence indicates membership in its complement. Hence, it might be expected that there are many links between reliable learning and classification.

Theorem 3.1 *Let $\mathcal{A}$ be learnable under a criterion which needs only finitely many mind changes, e.g.,* Ex *and* Ex^a.[2] *Then $\mathcal{A}$ is reliably learnable under this criterion iff $\mathcal{A}$ is one-sided.*

Proof ($\Rightarrow$): Let $\mathcal{A}$ be reliably learnable. The classifier outputs 0 if the learner changes its mind and outputs 1 if there is no mind change. Whenever the learner converges to an index, then the classifier outputs only finitely many 0s and thus accepts the language. A reliable learner does not converge on computable sets which are not learned and thus the classifier is correct. On the other hand if the learner does not converge and makes infinitely many mind changes, then the classifier also outputs infinitely many 0s and rejects the set on the input.

($\Leftarrow$): If $\mathcal{A}$ is learnable and one-sided classifiable, then a mind change can be introduced into the learning algorithm by padding at every place where the classifier outputs 0, i.e., if the learner outputs for σ and σw the same guess e, but the classifier outputs a 0 for σw, then the learner's output at σw is replaced by an equivalent but different index for the characteristic function computed by e. This does not effect convergence on $A \in \mathcal{A}$ since there these new mind changes are inserted only finitely often. But if $A \notin \mathcal{A}$, then the classifier outputs

[2] Ex^a-identification [12] requires that a *final* program p be output *and* that that p compute the input characteristic function with not more than a mistakes. Note that $\text{Ex} = \text{Ex}^0$.

infinitely many 0s which induce infinitely many mind changes on the modified learner; so this modified learner diverges and the modified learner is reliable, i.e., it converges on a computable A if and only if it learns A. ∎

Barzdins and Freivalds [6] introduced the notion of bounded mind changes where a machine has the right to output only a fixed finite number of guesses such that the last one of them is correct. This notion was more generally considered by Case and Smith [12]. Freivalds and Smith [13] generalized this concept further by using constructive ordinal [26] bounds. Their more general version of bounded mind changes is equivalent to the following notion of *well-bounded mind changes*.

Consider a recursively enumerable and well-ordered set $\{q_0, q_1, \ldots\}$ of rational numbers; well-ordered means that there is no infinite descending sequence $q_{i_0}, q_{i_1}, \ldots$, i.e., no sequence with $q_{i_{k+1}} < q_{i_k}$ for all k. An inductive inference machine M has *well-bounded mind changes* iff there is such a recursively enumerable well-ordered set $\{q_0, q_1, \ldots\}$ of rationals and M outputs for each hypothesis e_k also an associated rational q_{i_k} such that for every mind change from e_k to e_{k+1} the relation $q_{i_{k+1}} < q_{i_k}$ holds. This notion gives a sufficient but not necessary condition for two-sided classification.

Theorem 3.2 *If $\mathcal{A}$ can be reliably learned by a machine with well-bounded mind changes, then $\mathcal{A}$ is two-sided classifiable.*

Proof Let M be a machine with well-bounded mind changes which reliably infers $\mathcal{A}$. Since M diverges on input not in $\mathcal{A}$, there is at least one mind change such that $q_{i_{k+1}} \geq q_{i_k}$ at this k-th mind change. So M diverges iff there is a mind change with $q_{i_{k+1}} \geq q_{i_k}$ and a classifier just outputs 1 as long as the q_{i_k} form a strictly descending sequence and changes to 0 if a mind change with $q_{i_{k+1}} \geq q_{i_k}$ occurs. ∎

The condition of reliability in Theorem 3.2 is very restrictive since it enables one to construct a classifier with at most one mind change. So one would like to look for a more general sufficient condition. The next theorem replaces, then, reliable inference by Popperian Explanatory-identification (PEx), i.e., Ex-identification where every conjecture ever issued by the learner is an index for a total function [10, 12].

Theorem 3.3 *If $\mathcal{A}$ can be* PEx*-identified with a well-bounded number of mind changes, then $\mathcal{A}$ is two-sided classifiable.*

The proof of this theorem is based on the idea of emulating the learning process and conjecturing 1 whenever the learner places a hypothesis which firstly coincides with the data seen so far and secondly the mind change bound is not yet violated.

Nevertheless it turns out that both theorems have false converses. Indeed Theorem 3.4 shows that there is a two-sided classifiable class which cannot be Ex-learned with well-bounded mind changes – even in the absence of any further restriction.

Theorem 3.4 *There is a two-sided classifiable class $\mathcal{A} \in$* Ex *which cannot be* Ex*-learned with a well-bounded number of mind changes.*

Proof A *simple* set [27] is one which is recursively enumerable and whose infinite complement does not contain any infinite recursive set. Let $S = \{a_0, a_1, \ldots\}$ be a simple set and $\mathcal{A} = \{A : |A| \text{ is finite and even and } A \subseteq \overline{S}\}$. A two-sided classifier on input σ checks first whether $\sigma(a_k) = 1$ for some $a_k \in dom(\sigma)$ with $k \leq |\sigma|$. If so, then the classifier outputs 0. Otherwise the output is 1 if the number of all x with $\sigma(x)\downarrow = 1$ is even and is 0 if this number is odd. Since no infinite computable set is disjoint from S, this two-sided classifier for $\mathcal{A}$ is correct.

On the other hand $\mathcal{A}$ cannot be learned by well-bounded mind changes: Let M be an inductive inference machine which learns $\mathcal{A}$ with well-bounded mind changes. For each set $A \in \mathcal{A}$ let $q(A)$ be the minimal q_i output during the inference of A. The set $\{q(A) : A \in \mathcal{A}\}$ has a minimum q_j since it is well-ordered. $q_j = q(A)$ for some fixed set A. Now A has finite and even cardinality and there is some $\sigma \preceq A$ such that $M(\sigma)$ is an index for A and M has output q_j while reading this σ. Since $\overline{S}$ is infinite there are $x, y \in \overline{S} - A - dom(\sigma)$ and M has to infer $A \cup \{x, y\}$. Since also $\sigma \preceq A \cup \{x, y\}$, M has to make a mind change after σ and also output a rational $q_i < q_j$. So $q(A \cup \{x, y\}) < q_j$ in contradiction to the choice of q_j and such a machine M does not exist. ∎

A further connection between well-bounded learning on the one hand and classification on the other is the following which does not need any further assumption such as reliability or Popperian identification (PEx).

Theorem 3.5 *If $\mathcal{A}$ can be* Ex*-learned with well-bounded mind changes, then $\overline{\mathcal{A}}$ is one-sided.*

Theorem 3.5 needs the well-bound on the mind changes. In the unbounded case the class of all cofinite sets is one-sided and Ex-identifiable, but not two-sided. So there is an Ex-identifiable class without one-sided classifiable complement.

4 Classification From Only Positive Data

Gold [16] introduced the notion of identification from text. A text is a form of input where every set is presented as an sequence of numbers and the symbol "#", which contains each element of A at least once and which contains no numbers outside A. Analogously to Gold's notion of inference, classification from text is defined: a classifier reads more and more of a text of some set A and converges to 1 iff $A \in \mathcal{A}$. As in the case of standard classification, there are the obvious variants of one-sided and two-sided classification from text.

Example 4.1 *Every class $\mathcal{F}_\phi$ of all languages satisfying the Boolean formula ϕ is two-sided classifiable from text.*

Proof The classifier is relatively easy and for each input σ evaluates $\phi(range(\sigma))$. Since ϕ accesses the set A only at a finite number of places, all sufficient long $\sigma \preceq T$ for a given text T satisfy $x \in range(\sigma) \Leftrightarrow x \in A$ for the x where ϕ evaluates $A(x)$. E.g., if $\phi(A) = (3 \in A \wedge 4 \notin A)$, then all sufficiently large $\sigma \preceq T$ satisfy $3 \in range(\sigma) \Leftrightarrow 3 \in A$ and $4 \in range(\sigma) \Leftrightarrow 4 \in A$. So the result of evaluating ϕ on $range(\sigma)$ for these σ is the same as for evaluating ϕ on A. ∎

Theorem 4.2 *If $\mathcal{A}$ and $\mathcal{B}$ are both two-sided classifiable from text and a finite set belongs to $\mathcal{A}$ iff it belongs to $\mathcal{B}$, then $\mathcal{A} = \mathcal{B}$.*

Proof Assume that $\mathcal{A}$ and $\mathcal{B}$ are both two-sided classifiable from text, that each finite set belongs to $\mathcal{A}$ iff it belongs to $\mathcal{B}$ and that A is an infinite and computable set. Furthermore, let M_1 classify $\mathcal{A}$ and M_2 classify B from text and let $a_0, a_1, \ldots$ be a recursive enumeration of A. Now define inductively over k a text $T = a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots$ such that $M_1(a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots a_k\#^{n_k}) = M_2(a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots a_k\#^{n_k})$ for all k; the numbers n_k must all exist since M_1 and M_2 classify each finite set $\{a_0, a_1, a_2, \ldots, a_k\}$ in the same way and thus converge on each text $a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots a_k\#^{\infty}$ to the same value. So both, M_1 and M_2, take on T infinitely often the same value and both converge on T; therefore both converge to the same limit-value and A is in $\mathcal{A}$ iff A is in $\mathcal{B}$. ∎

One might ask whether the following chain-condition on two-sided classifiable $\mathcal{A}$ must hold.

Whenever an ascending chain $A_0 \subset A_1 \subset \ldots$ belongs to $\mathcal{A}$ so does some infinite set.

It needn't as as the following counterexample $\mathcal{A}$ shows. Let

$$\mathcal{A} = \{A : A \cap S = \emptyset\},$$

where S is a simple set, i.e., a recursively enumerable set whose infinite complement does not contain any infinite recursive set. This set S has a recursive enumeration $a_0, a_1, \ldots$ and the two-sided classifier M just checks whether the text seen so far intersects an approximation of S:

$$M(\sigma) = \begin{cases} 0 & \text{if } a_k \in range(\sigma) \text{ for some } k \leq |\sigma|; \\ 1 & \text{otherwise.} \end{cases}$$

Now let $\overline{S} = \{b_0, b_1, \ldots\}$ (where the sequence $b_0, b_1, \ldots$ is of course not computable). Then $\{b_0\}, \{b_0, b_1\}, \{b_0, b_1, b_2\}, \ldots$ forms this ascending chain of sets in $\mathcal{A}$. But $\mathcal{A}$ has no infinite member since every infinite and computable set intersects S; non-computable members of $\mathcal{A}$ are not considered in this paper.

Furthermore, Theorem 4.2 does not hold for one-sided classification. An example is $\mathcal{A}$ as the class of all finite sets and $\mathcal{B}$ as the class of all sets. Obviously $\mathcal{B}$ can be classified one-sided from text by always outputting 1. For $\mathcal{A}$ the algorithm is a bit more difficult: $H(\lambda) = 1$ and $H(\sigma w)$ is 1 if $w \in range(\sigma)$ and 0 if $w \notin range(\sigma)$. Thus if the text is for an infinite set, then infinitely often a new element is added and so H outputs infinitely often a 0. If the text is for a finite set, then only finitely often w is a new element and so the classifier converges to 1.

Theorem 4.3 *If $\mathcal{A}$ is one-sided classifiable from text and contains only infinite languages, then $\mathcal{A}$ is void. In particular the class $\mathcal{P}$ of all pattern-languages is not classifiable from text.*

Proof Let H be a classifier for $\mathcal{A}$ and $A = \{a_0, a_1, \ldots\}$ be an infinite set. Then there is a text $T = a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots$ such that $H(a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots a_k\#^{n_k}) = 0$ for all k since H must output on each text $a_0\#^{n_0}a_1\#^{n_1}a_2\#^{n_2}\ldots a_k\#^{\infty}$ for each finite set $\{a_0, a_1, \ldots, a_k\}$ infinitely many often a 0. So each set A has a text T such that H outputs on T infinitely many 0s.

The adaption to $\mathcal{P}$ uses the fact that every pattern language which contains two different elements already is infinite. Thus the construction starts with $a_0a_1\#^{n_1}$ and then proceeds in the same way. ▮

Indeed the construction can be strengthened to prove the existence of some kind of locking-set: If $\mathcal{A}$ can be one-sided classified from text and if $A \in \mathcal{A}$ is infinite, then there is a finite set $F \subseteq A$ such that every computable set B with $F \subseteq B \subseteq A$ belongs to $\mathcal{A}$.

Definition 4.4 A machine H classifies a class $\mathcal{A}$ *partially* iff H on any text T for any set A outputs an infinite sequence of numbers such that $A \in \mathcal{A}$ iff exactly one number appears in the output infinitely often and $A \notin \mathcal{A}$ iff no number appears in the output infinitely often.

It is easy to see that every class which can be one-sided classified from text can also be partially classified from text. But there are classes which can be partially classified but cannot be one-sided classified from text.

Theorem 4.5 *If $\mathcal{A}$ is a uniformly recursive family $A_0, A_1, \ldots$, then $\mathcal{A}$ can be partially classified.*

Proof W.l.o.g. for every $A \in \mathcal{A}$ there is exactly one e with $A = A_e$. The algorithm H outputs each number e on text $T = w_0w_1\ldots$ for A at least n times iff A_e and T are "compatible at level n", i.e., iff $\{w_0, w_1, \ldots, w_n\} \subseteq A_e$ and each $x \in A_e$ with $x \leq n$ appears in T.

On one hand if the set A to be classified equals A_e, then H outputs e infinitely often. On the other hand if $A \neq A_e$, then there is an n such that either $w_n \in A - A_e$ or $n \in A_e - A$. In both cases, H outputs e less than n times. In the first case $A \in \mathcal{A}$ and there is an unique index e such that H outputs e infinitely often. In the second case $A \notin \mathcal{A}$ and H outputs no e infinitely often. ▮

Since the classes $\mathcal{C}$, $\mathcal{D}$, $\mathcal{E}$, $\mathcal{G}$, $\mathcal{P}$ and $\mathcal{R}$ (from Example 2.4) are uniformly recursive families, they can be partially classified. Furthermore, all classes $\mathcal{F}_\phi$ (from Example 2.4) can be partially classified since they are two-sided classifiable from text.

Theorem 4.6 *If $\mathcal{A}$ is one-sided classifiable from informant, then $\mathcal{A}$ is partially classifiable from text. The converse does not hold.*

5 Structural Properties of Classification

Soare [27] contains an extensive study on the relation between recursively enumerable and computable sets. As Stephan [28] has already noted, the situation of one-sided versus two-sided classification is similar of that of recursively enumerable versus computable sets. This relationship does not only hold in the setting of classifying all sets but also in setting of the present paper of classifying computable sets.

This section shows that if only computable sets are to be classified, then the analogy with recursively enumerable versus computable sets is even more striking. Turing degrees, an important tool for studying recursively enumerable sets, are also found to be useful in analyzing the complexity of one-sided classification. The next result shows that – similarly to Stephan's general setting [28] – every one-sided class is two-sided relative to a sufficiently complex oracle.

An oracle U is *Turing reducible* to V (written: $U \leq_T V$) iff U can be computed by a machine which has access to a database containing V by the membership-queries "Is $x \in V$?". For an oracle U the relativized halting problem U' to U is defined as $U' = \{e : \varphi_e^U(e) \downarrow\}$.[3] U is *high* iff $K' \leq_T U'$ [4]. An alternative characterization is that there is a function u computable relative to U which dominates every recursive function, i.e., which satisfies $(\forall^\infty x)\,[u(x) > f(x)]$ for all $f \in$ REC. Adleman and Blum [1] showed that high oracles play a significant role in inductive inference: REC can be Ex-identifiable relative to U iff U is high. Theorems 5.1 and 5.4 show that the high oracles play a similar special role in classification.

Theorem 5.1 *For each high oracle U, every one-sided class $\mathcal{A}$ has a two-sided classifier which is computable relative to U.*

Proof Let H be a one-sided classifier for a class $\mathcal{A}$ of computable sets. Furthermore let u be a function computable relative to U which dominates every computable function. Now the two-sided classifier is defined as follows where $n_H(\sigma)$ denotes as in Theorem 2.2 the number of prefixes $\tau \preceq \sigma$ with $H(\tau) = 0$. The idea is now to repeat each 0 of H a large but finite number of times such that M still converges to 1 if H does but M converges to 0 if H only diverges.

If $u(n_H(\sigma)) > |\sigma|$, then let $M(\sigma) = 0$ else let $M(\sigma) = 1$.

If $A \in \mathcal{A}$, then there is only a finite number n of prefixes $\tau \preceq A$ with $H(\tau) = 0$. Almost all prefixes σ of A have length at least $u(n)$. So $|\sigma| \geq u(n) \geq u(n_H(\sigma))$ and $M(\sigma) = 1$ for these prefixes σ. If $A \notin \mathcal{A}$ and A is computable, then also the function $f_A(n) = \min\{m : n_H(A(0)A(1)\dots A(m)) \geq n\}$ is computable and thus u dominates f_A. There is a n with $u(m) > f(m)$ for all $m \geq n$. In particular whenever a prefix $\sigma \preceq A$ has at least the length $u(n)$, then $u(n_H(\sigma)) > f_A(n_H(\sigma)) \geq |\sigma|$ and $M(\sigma) = 0$. So M converges on every computable set outside $\mathcal{A}$ to 0 and M is two-sided. ▮

A recursively enumerable set E is called *creative* [27, Definition II.4.3] iff there is an effective procedure which disproves for every e the hypothesis "$W_e = \overline{E}$" by a counterexample $f(e)$, i.e., either $f(e) \in \overline{E} - W_e$ or $f(e) \in W_e - \overline{E}$. The name "creative" derives from the fact that such an f creates a new element $f(e) \in \overline{E}$ outside W_e whenever $W_e \subseteq \overline{E}$. This concept is adapted to the context of classifying computable sets.

Definition 5.2 A one-sided classifiable class $\mathcal{A}$ is *creative* iff there is an uniformly computable array $A_0, A_1, \dots$ such that for each one-sided classifier H_e the set A_e is a counterexample to the hypothesis "H_e classifies $\overline{\mathcal{A}}$".

The next theorem shows that there is a creative class, namely the class of all cofinite sets. So this class is effectively not two-sided.

Theorem 5.3 *The class $\mathcal{C}$ of all cofinite sets is creative.*

Proof Let inductively $A_e(0) = 0$ and $A_e(n+1) = H_e(A_e(0)A_e(1)\dots A_e(n))$. If H_e converges on this set to 1, then it is cofinite and not in $\overline{\mathcal{C}}$. Otherwise H_e does not converge to 1 and outputs infinitely many 0. Then also A_e is coinfinite and belongs to $\overline{\mathcal{C}}$. So A_e proves that H_e is not an one-sided classifiers for the complement of $\mathcal{C}$. $\mathcal{C}$ is creative since its complement is effectively not one-sided. ▮

All creative sets are 1-equivalent to K and have in particular the same Turing degree as K, i.e., belong to the greatest recursively enumerable Turing degree. So it is natural to ask how complex the creative classes are and the next theorem states, that there is indeed an analog result and that only the high oracles allow classifying them two-sided.

Theorem 5.4 *Every creative class, in particular $\mathcal{C}$, is two-sided only relative to high oracles.*

Proof It is easy to code an infinite array of machines $H_{s(e)}$ such that the machines are independent on the actual input A and that $H_{s(e)}$ outputs on any input A infinitely many 0s iff W_e is infinite. This can be achieved

[3] φ_e^U is the e-th partial recursive in U function.

[4] This differs slightly from Soare's definition [27, Definition IV.4.2]: Soare defined "$K' \equiv_T U'$" instead of "$K' \leq_T U'$" since he considers only oracles $U \leq_T K$.

easily by

$$H_{s(e)}(\sigma) = \begin{cases} 0 & \text{if } W_{e,|\sigma|+1} \neq W_{e,|\sigma|}; \\ 1 & \text{otherwise, i.e., if } W_{e,|\sigma|+1} = W_{e,|\sigma|}. \end{cases}$$

So if W_e is finite, then $H_{s(e)}$ suggests the class to contain all computable sets; and if W_e is infinite, then $H_{s(e)}$ suggests the class to contain no computable set. Thus in the first case, the counterexample has to be outside $\mathcal{B}$ and in $\mathcal{A}$ and in the second case, the counterexample has to be in $\mathcal{B}$ and so outside $\mathcal{A}$. If now some machine M classifies $\mathcal{A}$ two-sided, then M classifies in particular each set $A_{s(e)}$. It follows that

$$\begin{array}{llllll} W_e \text{ is finite} & \Rightarrow & A_{s(e)} \in \mathcal{A} & \Rightarrow & M \text{ converges on } A_{s(e)} \text{ to } 1; \\ W_e \text{ is infinite} & \Rightarrow & A_{s(e)} \notin \mathcal{A} & \Rightarrow & M \text{ converges on } A_{s(e)} \text{ to } 0. \end{array}$$

So using M it can be computed in the limit whether W_e is finite or infinite and thus the Turing degree of M must be high. ∎

While the preceding results mainly dealt with creative classes, this one deals with several degrees of non-creativeness. First it is shown that there are one-sided classes of intermediate complexity: they are two-sided relative to some non-high oracle but not relative to the empty oracle. In particular they are also not creative by Theorem 5.4.

Theorem 5.5 *For each $U \geq_T K$ which is also enumerable relative to K there is a class $\mathcal{A}$ such that a Turing degree contains a classifier for $\mathcal{A}$ iff U is computable relative to its jump. In particular there are intermediate one-sided classes; these are neither two-sided nor creative.*

There are two kinds of immunity-properties for classes:

- For a class $\mathcal{A}$ there is no uniformly computable array $A_0, A_1, \ldots$ of pairwise different sets such that $\{A_0, A_1, \ldots\} \subseteq \mathcal{A}$.
- No infinite two-sided class $\mathcal{B}$ is contained in $\mathcal{A}$.

The following theorems investigate the extent to which one-sided classes and their complements satisfy these requirements. But, the first result shows that a one-sided class and its complement can never be simultaneously immune.

Theorem 5.6 *For every one-sided class $\mathcal{A}$ there is an uniformly recursive array $A_0, A_1, \ldots$ of pairwise distinct sets such that the class $\mathcal{B} = \{A_0, A_1, \ldots\}$ is two-sided and either $\mathcal{B} \subseteq \mathcal{A}$ or $\mathcal{B} \subseteq \overline{\mathcal{A}}$. Furthermore there is a two-sided infinite class $\mathcal{A}$ which does not contain such a subclass $\mathcal{B}$.*

The next theorem states that there is something analogous to simple sets which are recursively enumerable and coinfinite but intersect every infinite computable set.

Theorem 5.7 *There is an infinite one-sided class such that its complement has no two-sided infinite subclass.*

Proof Let U be a set which is enumerable relative to K but whose infinite complement does not have an infinite K-recursive subset, i.e., U is a set which is simple relative to K. Now the class $\{A : A \cap U \neq \emptyset\}$ has an infinite complement but is not disjoint from any infinite two-sided class. ∎

It is well-known that every infinite recursively enumerable set has an infinite computable subset. Stephan [28] showed that this easy observation does not generalize to one-sided classification versus two-sided in his model which requires correct classification of non-computable sets. Since the classification of only computable sets is more well-behaved, the following problem might still have a positive solution.

Problem Does every infinite one-sided class have an infinite two-sided subclass?

6 Classification By Finding Trial-And-Error Programs

Baliga, Case, Jain, Sharma and Suraj studied in several papers [3, 4, 11] the concept of learning (or using) limiting or mind-changing programs (equivalently, K-recursive programs) instead of ordinary programs for classes of computable functions. This concept transfers quite naturally to classification: Instead of guesses 0 and 1, the classifier produces a sequence of programs such that each of this program converges in the limit to either 0 or 1 which then stands for the guess of the classifier. More formally such a classifier assigns to every input σ a primitive recursive program e such that $L(e) = \lim_n \varphi_e(n)$ exists and is either 0 or 1. As in inductive inference there are two notions of convergence.

- *Ex-style classification*: For every computable set A, the classifier outputs for almost all $\sigma \preceq A$ the same guess e and $L(e) = \mathcal{A}(A)$.
- *BC-style classification*: For every computable set A, the classifier outputs for almost all $\sigma \preceq A$ an index e_σ such that $L(e_\sigma) = \mathcal{A}(A)$.

Theorem 6.1 *Ex-style classification and two-sided classification coincide.*

Proof It is easy to see that outputting a constant 0 or 1 can be transferred into outputting a program which converges in the limit to 0 or 1, respectively. So only the direction to transfer an Ex-style classifier into an two-sided classifier for the same class is interesting. Given an Ex-style classifier M the new two-sided classifier N is defined by $N(\sigma) = \varphi_{M(\sigma)}(|\sigma|)$. Since M always outputs indices of primitive recursive functions, N is total. Assume now that A is computable. Then M outputs for almost all $\sigma \preceq A$ the same index e. Furthermore $\varphi_e(n) = \mathcal{A}(A)$ for almost all n. It follows that $N(\sigma) = \mathcal{A}(A)$ for almost all $\sigma \preceq A$. ▮

Theorem 6.2 *Every one-sided class has a BC-style classifier.*

Proof By Theorem 5.1 every one-sided class is classifiable two-sided relative to a high oracle, in particular it has a K-recursive classifier M. By the Limit-Lemma there is a primitive recursive function N such that $M(\sigma) = \lim_x N(\sigma, x)$. Using the substitution-theorem there is a primitive recursive procedure assigning to each σ and index $e(\sigma)$ for the function $f(x) = N(\sigma, x)$. This index $e(\sigma)$ is then the output of the BC-style classifier which classifies the same sets as M. ▮

It is easy to see that the concept of BC-style classification is closed under complementation. Thus the inclusion of one-sided classification into a BC-style classifier is proper. The proof showed that every class which is two-sided relative to the oracle K is already BC-style classifiable. This can be extended to a characterization of BC-style classification by the following theorem.

Theorem 6.3 *For a class $\mathcal{A}$ of computable sets the following is equivalent:*
(a) *$\mathcal{A}$ is BC-style classifiable.*
(b) *$\mathcal{A}$ is two-sided relative to K.*
(c) $\{e : \varphi_e$ *computes some* $A \in \mathcal{A}\} \leq_T K'$.
Here "φ_e computes A" means that φ_e is total, $\varphi_e(x)\downarrow = 1$ for each $x \in A$ and $\varphi_e(x)\downarrow = 0$ for each $x \notin A$.

References

[1] Leonard Adleman and Manuel Blum: Inductive Inference and Unsolvability. *Journal of Symbolic Logic* 56 (1991) 891–900.

[2] Dana Angluin: Inductive Inference of Formal Languages from Positive Data, *Information and Control* 45 (1980) 117–135.

[3] Ganesh Baliga and John Case: Learning with Higher Order Additional Information. *Proceedings of the Fifth International Workshop on Algorithmic Learning Theory, Reinhardsbrunn*, (1994) 64–75.

[4] Ganesh Baliga, John Case, Sanjay Jain and Mandayam Suraj: Machine Learning of Higher Order Programs. *Journal of Symbolic Logic*, 59 (1994) 486–500.

[5] Shai Ben-David: Can Finite Samples Detect Singularities of Read-Valued Functions? *Proceedings of the 24th Annual ACM Symposium on the Theory of Computer Science, Victoria, B.C.*, (1992) 390–399.

[6] Janis Barzdins and Rūsiņš Freivalds: Prediction and Limiting Synthesis of Recursively Enumerable Classes of Functions. In: *Theory of Algorithms and Programs, Vol. 1* (Latvian State University, edited by J. M. Barzdins, Riga 1974) 112–128 (in Russian).

[7] Lenore Blum and Manuel Blum: Toward a mathematical Theory of Inductive Inference. *Information and Control*, 28 (1975) 125–155.

[8] J. Richard Büchi: On a decision method in restricted second order arithmetic. In *Proceedings of the International Congress on Logic, Methodology and Philosophy of Science*, Standford University Press, Standford, California, 1960.

[9] J. Richard Büchi and Lawrence H. Landweber: Definability in the Monadic Second Order Theory of Successor. *Journal of Symbolic Logic* 34 (1969) 166–170.

[10] John Case, Sanjay Jain and Suzanne Ngo Manguelle: Refinements of Inductive Inference by Popperian and Reliable Machines. *Kybernetika*, 30 (1994) 23–52.

[11] John Case, Sanjay Jain and Arun Sharma: On Learning Limiting Programs. *International Journal of Foundations of Computer Science*, 3 (1992) 93–115.

[12] John Case and Carl H. Smith: Comparison of Identification Criteria for Machine Inductive Inference. *Theoretical Computer Science* 25 (1983) 193–220.

[13] Rūsiņš Freivalds and Carl H. Smith: On the Role of Procrastination for Machine Learning", *Information and Computation* 107 (1993) 237–271.

[14] William Gasarch, Mark Pleszkoch, Frank Stephan and Mahendran Velauthapillai: Classification Using Information. *To appear in: Annals of Mathematics and Artificial Intelligence.*

[15] William Gasarch, Mark Pleszkoch and Mahendran Velauthapillai: Classification Using Information – Conference Version. *Proceedings of the Fifth International Workshop on Algorithmic Learning Theory, Reinhardsbrunn*, (1994) 290–300, Lecture Notes in Artificial Intelligence, 872, Springer Verlag, Heidelberg, 1994.

[16] E. Mark Gold: Language Identification in the Limit. *Information and Control*, 10 (1967) 447–474.

[17] Klaus Peter Jantke: Monotonic and Non-Monotonic Inductive Inference. *New Generation Computing* 8 (1991) 349–360.

[18] Kevin Kelly: *The Logic of Reliable Inquiry.* Oxford University Press, Oxford, to appear.

[19] Ker-I Ko: *Complexity Theory of Real Functions.* Birkhäuser, Boston, 1991.

[20] Lawrence H. Landweber: Decision Problems for ω-Automata. *Mathematical Systems Theory*, 3 (1969) 376–384.

[21] Robert McNaughton: Testing and Generating Infinite Sequences by a Finite Automaton. *Information and Control* 9 (1966) 434–448.

[22] Eliana Minicozzi: Some Natural Properties of Strong Identification in Inductive Inference. *Theoretical Computer Science* 2 (1976) 345–360.

[23] Maurice Nivat and Dominique Perrin (editors): *Automata on Infinite Words.* Lecture Notes to Computer Science 192, Springer-Verlag, Heidelberg, 1984.

[24] Piergiorgio Odifreddi: *Classical recursion theory.* North-Holland, Amsterdam, 1989.

[25] Daniel N. Osherson, Michael Stob and Scott Weinstein: *Systems that learn.* Bradford / MIT Press, London, 1986.

[26] Gerald E. Sacks: *Higher Recursion Theory*, Perspectives in Mathematical Logic, Springer-Verlag, Heidelberg, 1990.

[27] Robert I. Soare: *Recursively enumerable sets and degrees.* Springer-Verlag, Heidelberg, 1987.

[28] Frank Stephan: On One-Sided Versus Two-Sided Classification. Manuscript.

[29] Alan M. Turing: On Computable Numbers with an Application to the Entscheidungsproblem. *Proc. London Math. Society* 42 (1936) 230–265.

[30] Rolf Wiehagen and Carl H. Smith: Generalization versus Classification. *Journal of Experimental and Theoretical Artificial Intelligence*, 7, 1995. Shorter version in *Proceedings 5th Annual Workshop on Computational Learning Theory*, (1992) 224–230. ACM Press, New York.

[31] Carl H. Smith, Rolf Wiehagen and Thomas Zeugmann: Classifying Predicates and Languages. *To appear in:* International Journal of Foundations of Computer Science.

[32] Boris A. Trakhtenbrot: Finite Automata and the Logic of One Place Predicates. *Sibirian Mathematical Journal* 3 (1962) 103–131 [in Russian].

[33] Thomas Zeugmann: Characterizations of Monotonic and Dual Monotonic Language Learning, *Information and Computation* 120 (1995) 155–173.

[34] Thomas Zeugmann and Steffen Lange: A Guided Tour Across the Boundaries of Learning Recursive Languages. *Algorithmic Learning for Knowledge-Based Systems* (K. P. Jantke and S. Lange, Eds.), Lecture Notes in Computer Science 961 (1995) 193–262.