Classifying Recursive Predicates and Languages

Rolf Wiehagen*
Universität Kaiserslautern
Fachbereich Informatik
P.O. Box 3049
67653 Kaiserslautern, Germany
wiehagen@informatik.uni-kl.de

Carl H. Smith†
Department of Computer Science
University of Maryland
College Park,
MD 20742 USA
smith@cs.umd.edu

Thomas Zeugmann
TH Darmstadt
Institut für Theoretische Informatik
Alexanderstr. 10
64283 Darmstadt, Germany
zeugmann@iti.informatik.th-darmstadt.de

Abstract

We study the classification of recursive predicates and languages. In particular, we compare the classification of predicates and languages with the classification of arbitrary recursive functions and with learning. Moreover, we refine our investigations by introducing classification with a bounded number of mind changes and establish a new hierarchy. Furthermore, we introduce *multi–classification* and characterize it. Finally, we study the classification of families of languages that have attracted a lot of attention in learning theory.

1. Introduction

Learning and *classification* have attracted considerable attention by computer scientists, both in theory and practice. *Inductive inference* is an important aspect of learning that has been widely studied (cf. Angluin and Smith (1983, 1987)). The inductive inference problem is to take finite samples of some target concept and to

*The first author was supported by the German Ministry for Research and Technology (BMFT) under grant no. 01 IW 101 E7

†The second author was supported in part by NSF Grant 9020079

generalize an algorithm that can produce all other samples of the same concept. Hence, inductive inference may be regarded as the most general framework to study the *generalization problem* (cf. Michalski et al. (1983)). The *classification problem* may be described as follows: Given a number of, usually finite, choices, one takes finite samples of a target concept and has to find out algorithmically to which of the possible choices the concept belongs to (cf. Duda and Hart (1973)).

Recently, the problem of classification has been compared with the inductive inference problem in a recursion theoretic setting (cf. Wiehagen and Smith (1992)). In that paper a new formalization of classification was introduced.

The aim of the present paper is fourfold. First, we apply the previously developed formalism to investigate the classification of $\{0,1\}$ valued recursive functions. These functions are often called *predicates* as they represent binary decisions on the input. By utilizing the isomorphism between strings of symbols and the natural numbers, the predicates over the natural numbers also represent *formal languages*. Hence, we simultaneously study the classification of predicates and languages. On the one hand, we are interested in learning the differences and similarities between the classification of predicates and arbitrary recursive functions. On the other hand, we enrich this study by considering the following two cases. In the first case we are satisfied if the classification algorithm produces any *one* of the possibly many correct answers. In the second case we require the classification algorithm to produce *all* of the correct classifications (cf. Definition 3). Second, we compare the power of classification algorithms that are allowed to produce only a single guess (finite classification, cf. Definition 2, the $c = 0$ case) with those that may change their mind a predetermined fixed number of times (cf. Definition 2, the $c \in \mathbb{N}$ case). Third, we introduce the notion of *consistent* classification (cf. Definition 4) and compare it with general classification. Finally, we study classification of families of languages that have received considerable attention in learning theory.

2. Technical Preliminaries

By $\mathbb{N} = \{0, 1, 2, ...\}$ we denote the set of all natural numbers. Members of $\mathbb{N}$ will serve as names for programs. The function computed by program i will be denoted by φ_i. Most reasonable ways of assigning names to programs results in a list φ_0, φ_1, ... called an *acceptable programming system* (cf. Machtey and Young (1978)). By $\mathcal{R}$ we denote the class of all recursive functions. The class of $\{0,1\}$ valued recursive functions, our model of both predicates and languages, is denoted by $\mathcal{R}_{0,1}$. A set is *recursively enumerable* (r.e.) iff it is the domain of some φ_i. A more intuitive, equivalent characterization of the r.e. sets is to call a set r.e. if it is either the range of a recursive function or it is empty. In this way it is easy to see that the r.e. sets are the ones for which we can write a procedure that prints out, eventually, all and only members of the set in question. The i^{th} r.e. set is denoted by W_i. Subset is denoted by $\subseteq$ and $\subset$ denotes *proper* subset. The quantifier $\forall^\infty$ is interpreted as "all but finitely

many." For a function $f \in \mathcal{R}$ and $n \in \mathbb{N}$, let $f^n = \mathrm{cod}(f(0), f(1), ..., f(n))$, where cod denotes a computable and bijective mapping from the set $\mathbb{N}^*$ of all finite sequences of natural numbers onto $\mathbb{N}$. Sometimes, for the sake of simplicity of notation, we identify $\alpha \in \mathbb{N}^*$ with $\mathrm{cod}(\alpha)$, for α a finite function.

Whenever appropriate we shall represent a recursive function by the *sequence of its values*. For example, the sequence $0^2 1^3 2^\infty$ denotes the function

$$f(x) = \begin{cases} 0, & \text{if} \quad x < 2 \\ 1, & \text{if} \quad 2 \leq x < 5 \\ 2, & \text{otherwise} \end{cases}$$

Next, we formalize the models of identification and classification mentioned in the introduction. As in Gold (1967) we define an *inductive inference machine* (abbr. IIM) to be an algorithmic device which works as follows: The IIM takes as its input larger and larger initial segments of the graph of a function and it either requests the next function value, or it first outputs a hypothesis, i.e., a name of a program, and then it requests the next function value.

A classification machine (abbr. CM) takes as input the graph of a function (as IIMs do) and it either requests the next function value, or it first outputs an integer chosen ¿from a finite set, and then it requests the next function value.

Let M be an IIM or a CM. Furthermore, let i and j be two consecutive hypotheses produced by M. We say that M changes its mind, or synonymously, M performs a mind change, iff $i \neq j$. When dealing with mind changes, it is technically much more convenient to require the IIMs to behave as follows. Let f be a recursive function. If M on $f(0), ..., f(n)$ outputs its first guess, then it has to output a hypothesis at any subsequent step. It is easy to see that any IIM M may be straightforwardly converted into an IIM $\hat{M}$ behaving as required such that both machines produce the same sequence of mind changes.

We start with the formalization of learning. The following definition is due to Gold (1967) (the $c = 0$ case) and Barzdin (1971) and Blum and Blum (1975) (the $c = *$ case).

Definition 1. *Let $U \subseteq \mathcal{R}$ and let $c \in \mathbb{N} \cup \{*\}$. The class U is said to be learnable with at most c mind changes iff there is an IIM M such that for all $f \in U$*

(1) *there is a j such that $\varphi_j = f$ and $M(f^n) = j$ for almost all $n \in \mathbb{N}$,*

(2) *M, when successively fed $f(0)$, $f(1)$, ... performs at most c ($c = *$ means at most finitely many) mind changes, i.e., $card(\{n | M(f^n) \neq M(f^{n+1})\}) \leq c$.*

If U can be learned by an IIM M with at most c mind changes, then we write $U \in EX_c(M)$. The class of all sets of recursive functions that are learnable with at

most c mind changes is denoted by EX_c, an abbreviation for *explains* as a program for f can be regarded as an explanation of the set of examples constituting the graph of f (cf. Case and Smith (1983)). If U can be learned with 0 mind changes, then we also say that U is *finitely* learnable. Moreover, we set $FIN =_{df} EX_0$. If $c = *$, then we usually omit the lower index and simply say U can be learned.

Next, we formalize classification of finitely many sets.

Definition 2. *Let $S_0, ..., S_{k-1} \subseteq \mathcal{R}$ and let $S = S_0 \cup \ ... \ \cup S_{k-1}$. Furthermore, let $c \in \mathbb{N} \cup \{*\}$. Then $(S_0, ..., S_{k-1})$ is said to be classifiable with at most c mind changes iff there is a CM M such that for all $f \in S$*

(1) *for all $n \in \mathbb{N}$, whenever M, on input f^n, outputs a hypothesis j, then $j \in \{0, ..., k-1\}$,*

(2) *there is a j such that $f \in S_j$ and $M(f^n) = j$ for almost all $n \in \mathbb{N}$,*

(3) *M, when successively fed $f(0)$, $f(1)$, ... performs at most c ($c = *$ means at most finitely many) mind changes, i.e., $card(\{n | M(f^n) \neq M(f^{n+1})\}) \leq c$.*

If $(S_0, ..., S_{k-1})$ is classified by a CM with at most c mind changes, then we write $(S_0, ..., S_{k-1}) \in CL_k^c(M)$. By CL_k^c we denote the collection of all k–tuples of sets that are classifiable with at most c mind changes, i.e.,

$$CL_k^c = \{(S_0, ..., S_{k-1}) | \ \exists \text{CM } M \ [(S_0, ..., S_{k-1}) \in CL_k^c(M)]\}.$$

Moreover, we set $CL^c =_{df} \bigcup_{k \geq 2} CL_k^c$. If $c = *$, then we usually omit the upper index, and say simply that $(S_0, ..., S_{k-1})$ is classifiable. Furthermore, if $c = 0$, then we also say that $(S_0, ..., S_{k-1})$ is *finitely classifiable*, and set $FCL_k =_{df} CL_k^0$. In other words, a k–tuple $(S_0, ..., S_{k-1})$ is finitely classifiable if M's first guess is always a correct one. Finally, we set $FCL =_{df} \bigcup_{k \geq 2} FCL_k$.

In the definition above requirement (1) specifies the set of allowed hypotheses. Clearly, any guess not contained in $\{0, ..., k-1\}$ cannot be correct. Hence, it is only natural to restrict the hypothesis space to the set $\{0, ..., k-1\}$. Nevertheless, this requirement does not restrict the capabilities of CMs. This can be seen as follows. Suppose we are given a CM M classifying some k-tuple $(S_0, ..., S_{k-1})$ of sets that uses the set $\mathbb{N}$ as its hypothesis space. Then M may be converted into a CM $\hat{M}$ satisfying requirement (1) such that $(S_0, ..., S_{k-1}) \in CL_k(\hat{M})$. On any input, the CM $\hat{M}$ simply simulates M. If M outputs a guess from $\{0, ..., k-1\}$ then $\hat{M}$ behaves thus. Otherwise, $\hat{M}$ suppresses M's output and requests the next input. It is easy to prove that $CL_k^c(M) = CL_k^c(\hat{M})$. Consequently, when dealing with classification, class preserveness can be realized without loss of generality. On the other hand, looking at IIMs the situation changes considerably. The analog of requirement (1) reads as "for all $f \in U$, all $n \in \mathbb{N}$, whenever M on input f^n outputs a hypothesis j, then $\varphi_j \in U$." However, the demand to work class preservingly does seriously restrict the learning power of IIMs (cf. Jantke and Beick (1981)).

In the next definition we consider the situation that the sets $S_0, ..., S_{k-1}$ are not necessarily disjoint. Looking at potential applications, it might be highly desirable not to obtain only *one* index of a set the target function f belongs to, but the indices of *all* those sets that contain f. For example, consider the case of automated medical diagnosis. In this case, we would certainly desire to be aware of *all* the diseases that manifest the observed symptoms. In our formalism, each disease is set S_i and the functions to be classified map symptoms to *present* or *not present*.

Definition 3. *Let $S_0, ..., S_{k-1} \subseteq \mathcal{R}$, and let $S = S_0 \cup \ ... \ \cup S_{k-1}$. Furthermore, let $ALL = \{0, ..., k-1\}$. Then $(S_0, ..., S_{k-1})$ is said to be multi–classifiable iff there is a CM M such that for all $f \in S$*

(1) *for all $n \in \mathbb{N}$, whenever M, on input f^n, outputs a hypothesis HYP, then HYP $\subseteq$ ALL,*

(2) *there is a non-empty set $SUB \subseteq ALL$ such that*

   (a) $M(f^n) = SUB$ for almost all n,

   (b) $f \in S_j$ for all $j \in SUB$,

   (c) $f \notin S_m$ for all $m \in ALL \backslash SUB$.

If $(S_0, ..., S_{k-1})$ is multi–classified by a CM, then we write $(S_0, ..., S_{k-1}) \in$ *Multi–*$CL_k(M)$. By *Multi–*CL_k we denote the collection of all k–tuples of sets that are multi–classifiable. Furthermore, we set *Multi–*$CL = \bigcup_{k \geq 2}$*Multi–*CL_k.

Finally, we introduce *consistent* classification. The main intention is as follows. Since potential users of a CM M never know whether M has successfully finished its classification task, they might want to be sure that the actual hypotheses they receive do correctly reflect the information the CM has been fed with.

Definition 4. *Let $S_0, ..., S_{k-1} \subseteq \mathcal{R}$ and let $S = S_0 \cup \ ... \ \cup S_{k-1}$. Then $(S_0, ..., S_{k-1})$ is said to be consistently classifiable iff there is a CM M such that*

(1) $(S_0, ..., S_{k-1}) \in CL_k(M)$,

(2) *for all $f \in S$, and for all $n, i \in \mathbb{N}$, if $M(f^n) = i$, then there must be a function $g \in S_i$ such that f and g coincide up to n.*

If $(S_0, ..., S_{k-1})$ is consistently classifiable by a CM M then we write $(S_0, ..., S_{k-1}) \in$ *Cons–*$CL_k(M)$. By *Cons–*CL_k we denote the collection of all k–tuples of sets that are consistently classifiable. Finally, we set *Cons–*$CL =_{df} \bigcup_{k \geq 2}$ *Cons–*CL_k.

3. Classification of Predicates versus Classification of Arbitrary Functions

In this section we compare the classification of $\{0,1\}$ valued functions with the classification of functions in general. Looking at learning there are several results establishing major differences between the learnability of classes of predicates and the inferability of arbitrary classes of recursive functions (cf. Blum and Blum (1975), Zeugmann (1983, 1988), Osherson, Stob and Weinstein (1986)). Moreover, Freivalds, Kinber and Wiehagen (1992) discovered that consistent learning of predicates considerably differs from consistent identification of arbitrary recursive functions. Hence, it is only natural to ask whether there are differences between the classification of predicates and arbitrary recursive functions.

Clearly, if a collection of sets of $\{0,1\}$ valued functions is classifiable, then it is classifiable as a collection of sets of arbitrary recursive functions. To consider the converse direction, we need the following notation. Let $S \subseteq \mathcal{R}$; then we use $\rho(S)$ to denote the restriction of S to predicates, i.e., $\rho(S) = S \cap \mathcal{R}_{0,1}$.

Theorem 1. *There is a collection of pairwise disjoint sets S_0, S_1, S_2 such that*

(1) $\mathcal{R} = S_0 \cup S_1 \cup S_2$,

(2) $(S_0, S_1, S_2) \notin CL$,

(3) $(\rho(S_0), \rho(S_1), \rho(S_2)) \in CL_3^2$.

Proof. The desired splitting of $\mathcal{R}$ is defined as follows. Let $S_0 = \{f \mid f \in \mathcal{R},$ *range* f is infinite$\}$, $S_1 = \{f \mid f \in \mathcal{R},$ *range* f is finite, $\forall x[f(x) = \max$ *range* $f \rightarrow f(x+1) \neq f(x)]\}$, and $S_2 = \mathcal{R} \setminus (S_0 \cup S_1)$. Obviously, $\mathcal{R} = S_0 \cup S_1 \cup S_2$ and S_0, S_1, and S_2 are pairwise disjoint. It remains to show that assertions (2) and (3) are fulfilled.

Claim 1. $(S_0, S_1, S_2) \notin CL$.

Suppose the converse, i.e., there is a CM M such that $(S_0, S_1, S_2) \in CL_3(M)$. We continue with the definition of a recursive function f on which M fails. Define $f(0), f(1), \ldots$ to be zero until the least k is found such that M, when successively fed $f(0), \ldots, f(k)$, outputs its first hypothesis. Let $j = M(f^k)$. Now we distinguish between the following cases.

Case 1. $j = 0$ or $j = 1$.

Then set $f(k+1) = 0$.

Case 2. $j = 2$.

Then define $f(k+1) = 1$.

We proceed inductively as follows. Let k be the largest element for which f is already defined. Compute $j = M(f^k)$. We consider the following cases.

Case 1. $j = 0$. Then define $f(k+1) = 0$.

Case 2. $j = 1$.

Then set $f(k+1) = \max\{f(y) | y \leq k\}$.

Case 3. $j = 2$.

Then define $f(k+1) = \max\{f(y) | y \leq k\} + 1$.

Obviously, f is recursive. Therefore, there has to be $j \in \{0, 1, 2\}$ such that $M(f^n) = j$ for almost all n.

Case 1. $j = 0$.

Due to the definition of f we obtain that $f(x) = 0$ for almost all $x \in \mathbb{N}$. Hence, f has a finite range and therefore it does not belong to S_0, a contradiction.

Case 2. $j = 1$.

In accordance with our construction, *range* f is finite. However, $f(x) = \max\{f(y) | y \in \mathbb{N}\}$ for almost all $x \in \mathbb{N}$. Therefore, M again misclassifies f.

Case 3. $j = 2$.

By definition of f, *range* f is infinite. Hence, f belongs to S_0 and not to S_2. Consequently, M again fails to correctly classify f.

This proves Claim 1.

Claim 2. $(\rho(S_0), \rho(S_1), \rho(S_2)) \in CL_3^2$.

In order to see this, first observe that $\rho(S_0) = \emptyset$. Moreover, $\rho(S_1) = \{f | f \in \mathcal{R}_{0,1}, \forall x[f(x) = 1 \rightarrow f(x+1) = 0]\}$ and $\rho(S_2) = \{f | f \in \mathcal{R}_{0,1}, \exists x[f(x) = f(x+1) = 1]\} \cup \{0^\infty\}$. Now it is easy to define a CM M' witnessing $(\rho(S_0), \rho(S_1), \rho(S_2)) \in CL_3^2$. The machine M' initially conjectures "2" until (if ever) an input $f(x)$ with $f(x) = 1$ is received. This is the correct behavior on the constant zero function. If an input 1 is received, then the input is not the constant zero function, and M' conjectures "1." M' continues scanning its input looking for two consecutive inputs that are 1. If this happens, then M' changes its conjecture back to "2" and stops. ☒

The theorem has the following corollary.

Corollary 2. *For any $n \geq 3$ there is a collection of pairwise disjoint sets exhausting $\mathcal{R}$ that is not in CL_n, but the collection of their restrictions to predicates is in CL_n.*

Proof. The idea is to divide S_1 from the proof of Theorem 1 into $n-2$ sets. These are the sets of recursive functions with finite range where the largest element of the range occurs at most once in a row, at most twice in a row, at most three times in a row, etc. The proof of Theorem 1 is then easily extended. ☒

The latter results show that there might be interesting differences between the classification of predicates and arbitrary functions. Later on, we shall point out some more differences. In the next section we study the power of classification algorithms with respect to the allowed number of mind changes.

4. Finite Classification versus Classification and Learnability

Our next theorem states that even the classification of two sets might be too complex to be done by a finitely working CM.

Theorem 3. *There are pairwise disjoint sets S_0, S_1 such that*

(1) $S_0 \cup S_1 = \mathcal{R}_{0,1}$,

(2) $(S_0, S_1) \in CL \setminus FCL$.

Proof. Let S_0 be the class containing only the function 0^∞. To immediately satisfy (1) and (2) above, let $S_1 = \mathcal{R}_{0,1} \setminus S_0$. A CM M that classifies (S_0, S_1) initially outputs "0" and then waits for a non zero input. When, and if, this input arrives, M changes its mind to "1." Clearly, $(S_0, S_1) \in CL(M)$.

Suppose by way of contradiction that $\hat{M}$ is a CM such that $(S_0, S_1) \in FCL(\hat{M})$. Then there must be an n such that $\hat{M}$, on input 0^n, outputs "0" as its conjectured classification. If not, then $\hat{M}$ fails to classify 0^∞ as a member of S_0, a contradiction. On the other hand, $\hat{M}$ outputs, when successively fed 0^n1^∞, the hypothesis "0" as its first guess. Therefore, $\hat{M}$ cannot finitely distinguish 0^∞ from 0^n1^∞. Consequently, $\hat{M}$ fails to properly classify at least one of those functions. ☒

The above theorem has an easy extension to the following:

Theorem 4. *For any $k > 1$ there are pairwise disjoint sets $S_0, \cdots, S_{k-1}$ such that*

(1) $S_0 \cup S_1 \cup \cdots \cup S_{k-1} = \mathcal{R}_{0,1}$,

(2) $(S_0, S_1, \cdots, S_{k-1}) \in CL \setminus FCL$.

Proof. Let $k > 1$ be arbitrarily fixed. For $0 \leq i < k-1$, S_i is the subset of $\mathcal{R}_{0,1}$ that contains only functions f such that the cardinality of the set $\{x \mid f(x) \neq 0\}$ is exactly i. Moreover, we set $S_{k-1} = \mathcal{R}_{0,1} \setminus (S_0 \cup \cdots \cup S_{k-2})$. Clearly, (1) and (2) above are satisfied. A CM M that classifies $S_0, S_1, \cdots, S_{k-1}$ initially outputs "0" and then reads input, counting the number of nonzero inputs. After a nonzero input is observed, resulting in a total of j nonzero inputs, M outputs the minimum of $\{j, k-1\}$. Clearly, $(S_0, S_1, \cdots, S_{k-1}) \in CL_k(M)$.

Suppose by way of contradiction that M' is a CM such that $(S_0, S_1, \cdots, S_{k-1}) \in FCL_k(M')$. As in Theorem 3, there must be an n such that M', on input 0^n produces a conjecture "0." Then M' will improperly classify at least one of 0^∞ or 0^n10^∞. ☒

In contrast to Theorem 4, it is possible to split $\mathcal{R}_{0,1}$ into disjoint, finitely classifiable sets.

Theorem 5. *For any $k > 1$ there are pairwise disjoint sets $S_0, \cdots, S_{k-1}$ such that*

(1) $S_0 \cup \cdots \cup S_{k-1} = \mathcal{R}_{0,1}$,

(2) $(S_0, \cdots, S_{k-1}) \in FCL$.

Proof. Let $S_0 = \{f \mid f \in \mathcal{R}_{0,1} \text{ and } f(0) = 0\}$. For $1 \leq i < k$, $S_i = \{f \mid f \in \mathcal{R}_{0,1}, f(j) = 1 \text{ for } 0 \leq j < i \text{ and } f(i) = 0\}$. Finally, $S_{k-1} = \mathcal{R}_{0,1} \setminus (S_0 \cup S_1 \cup \cdots \cup S_{k-2})$. Clearly, (1) and (2) above hold. A CM M that finitely classifies $(S_0, S_1, \cdots, S_{k-1})$ starts by reading input until the values of $f(0)$, $f(1)$, $\cdots$, $f(k-2)$ are all known. If all of these values are 1, then output "$k-1$" as the only conjecture. If not, then output "i" where i is the least number such that $f(i) = 0$. ☒

Next we compare finite classification and learnability. If a classification machine outputs a hypothesis i, then we know almost *all* about the corresponding set S_i, since there are only finitely many sets $S_0, ..., S_{k-1}$. On the other hand, if an IIM outputs a hypothesis i, then we know almost *nothing* about the corresponding function φ_i. Hence, at first glance it seems much easier to disprove a CM than to fool an IIM. However, the situation is much more subtle, as the following theorems show.

Theorem 6. *For any $S_0 \subset \mathcal{R}_{0,1}$ such that $S_0 \in FIN$, there is a S_1 such that*

(1) $S_1 \subset \mathcal{R}_{0,1}$,

(2) $S_1 \in FIN$,

(3) $S_0 \cap S_1 = \emptyset$,

(4) $(S_0, S_1) \in FCL$.

Proof. Suppose $S_0 \subset \mathcal{R}_{0,1}$ and $S_0 \in FIN$. Since no set in FIN can contain an accumulation point (cf. Freivalds and Wiehagen (1979)), there is a finite, $\{0,1\}$ valued, initial segment α such that no function in $\mathcal{R}_{0,1}$ extending α is in S_0. Let $S_1 = \{\alpha 0^\infty, \alpha 1^\infty\}$. Clearly, $S_1 \subset \mathcal{R}_{0,1}$. That S_1 is in FIN is witnessed by a IIM that "knows" α, waits for it to be seen as input, and then after seeing one more input will be able to guess the correct function from the two choices. By the choice of α, $S_0 \cap S_1 = \emptyset$. A CM M witnessing $(S_0, S_1) \in FCL$ knows α. Let $\alpha = (\alpha_0, ..., \alpha_n)$. Then M requests more and more inputs until it has seen $(f(0), ..., f(n))$. In case $f^n = \alpha$ it outputs "1" and "0" otherwise. Clearly, $(S_0, S_1) \in FCL_2(M)$. ☒

The latter theorem allows the following interpretation. Any "easily" learnable class can be completed to an "easily" classifiable pair. Thus, sometimes learning and classification are *not* very distinct. On the other hand, we have the following:

Theorem 7. *For any $S_0 \subset \mathcal{R}_{0,1}$ such that $S_0 \in FIN$, there is a S_1 such that*

(1) $S_1 \subset \mathcal{R}_{0,1}$,

(2) $S_1 \notin EX$,

(3) $S_0 \cap S_1 = \emptyset$,

(4) $(S_0, S_1) \in FCL$.

Proof. Let α as in the proof of Theorem 6. Set $S_1 = \{\alpha f \mid f \in \mathcal{R}_{0,1}\}$. A CM witnessing $(S_0, S_1) \in FCL$ may work as described in the proof of Theorem 6. It remains to show that $S_1 \notin EX$. Suppose the converse, i.e., there is an IIM M such that $S_1 \in EX(M)$. Then any program for M can be used to construct an IIM $\hat{M}$ that infers $\mathcal{R}_{0,1}$, a contradiction (cf. Gold (1965)). This can be seen as follows. On any input f^n, $n \in \mathbb{N}$, the IIM $\hat{M}$ simulates M on input αf^n. Suppose $j = M(\alpha f^n)$. Then $\hat{M}$ outputs a canonical program of the function η defined as follows: $\eta(x) = \varphi_j(x+n)$ for all $x \in \mathbb{N}$. Clearly, $\eta = f$ if and only if $\varphi_j = \alpha f$. On the other hand, since $S_1 \in EX(M)$, there is a j such that $M(\alpha f^n) = j$ for almost all n. Consequently, $\hat{M}$ learns f. ☒

5. Bounding the Number of Mind Changes

For finite classification (FCL) a CM is not allowed to change its conjecture at all. For standard classification (CL) a CM is allowed to change its conjecture an arbitrary finite number of times. The precise number of mind changes has not to be determined in advance. In the study of inductive inference, a mind change hierarchy was discovered by fixing in advance a particular number of mind changes that an IIM was allowed to make (cf. Case and Smith (1983)). In the sequel we present a hierarchy for classification based on a fixed number of allowed mind changes.

Theorem 8. *For any k there are sets S_0, S_1, $\cdots$, S_{k+1} such that*

(1) $S_0 \cup S_1 \cup \cdots \cup S_{k+1} = \mathcal{R}_{0,1}$,

(2) $(S_0, \cdots, S_{k+1}) \in CL^{k+1} \setminus CL^k$.

Proof. Suppose k is given. For $i \leq k$, let S_i be the subset of $\mathcal{R}_{0,1}$ such that for any $f \in S_i$ the cardinality of $\{x \mid f(x) = 1\}$ is exactly i. For example, $S_0 = \{0^\infty\}$. Finally, $S_{k+1} = \mathcal{R}_{0,1} \setminus (S_0 \cup S_1 \cup \cdots \cup S_k)$. Clearly, (1) above is satisfied.

A CM that witnesses $(S_0, \cdots, S_{k+1}) \in CL^{k+1}$ may work as follows: Let f be any fixed predicate. The CM starts by conjecturing "0". If the most recent conjecture is "i," $i \leq k$ and another datum $f(x)$ that is 1 is observed as input, then the CM conjectures "$i+1$." When the CM produces a conjecture "i" it will constitute the i^{th} change of conjecture. Clearly, the CM described above is the desired witness.

The proof is completed by showing that $(S_0, \cdots, S_k) \notin CL^k$. Suppose by way of contradiction that M is a CM such that $(S_0, \cdots, S_k) \in CL^k(M)$. A finite $\{0,1\}$ valued function is constructed in at most $k+1$ steps below. It will turn out that some extension of this finite functions will be an $f \in \mathcal{R}_{0,1}$ that M cannot properly classify. The finite initial segment constructed prior to step s is denoted by the string σ^s. By way of initialization, σ^0 is the empty string. An invariant over steps $0 \leq s \leq k$ is that

after step s has been completed, the number of occurrences of "1" in σ^{s+1} is exactly s and M, when successively fed σ^{s+1}, will make exactly s conjectures.

> *Step* 0: Look for the least n such that M on input 0^n produces a conjecture. If such an n is found, set $\sigma^1 = 0^n$ and go to Step 1. If no such n exists, then M fails to correctly classify the everywhere zero function in S_0.
>
> *Step* s, for $0 < s \leq k$: Let σ^s be the initial segment already defined. Look for the first pair (n, m) in the Cantor enumeration of $\mathbb{N} \times \mathbb{N}$ such that $M(\sigma^s) \neq M(\sigma^s 0^m 1 0^n)$. In other words, we seek for one more mind change of M on some additional data $0^m 1 0^n$. If such an m and n are found then set $\sigma^{s+1} = \sigma^s 0^m 1 0^n$ and go to Step $s+1$ until $s+1 > k$.

First we prove by induction the invariant that after step s has been completed, the number of occurrences of "1" in σ^{s+1} is exactly s and M, when successively fed σ^{s+1}, will make exactly s conjectures. After Step 0 has been completed, σ^1 has no ones present. Moreover, M, on input σ^1, outputs its first guess. Suppose inductively that for $s < k$, after Step s has been completed, the number of occurrences of "1" in σ^{s+1} is exactly s and M, when successively fed σ^{s+1}, will make exactly s conjectures. After Step $s+1$ is complete, exactly one more "1" has been added to the string σ^{s+1}, and M has been forced to make one more mind change.

By the comments in Step 0, we know that Step 0 must be completed, as otherwise M fails to classify the function 0^∞. Suppose Step $s < k$ is the last step completed. Let $f_1 = \sigma^{s+1}0^\infty$ and $f_2 = \sigma^{s+1}10^\infty$. By the invariant over stages, $f_1 \in S_s$ and $f_2 \in S_{s+1}$. By the failure of Step $s+1$ to complete, M will classify both f_1 and f_2 as in the same set, a contradiction.

To complete the proof, suppose Step k completes. Let $f_1 = \sigma^{k+1}0^\infty$ and $f_2 = \sigma^{k+1}10^\infty$. By the invariant over stages, $f_1 \in S_k$ and $f_2 \in S_{k+1}$. M will make its full allotment of mind changes on the initial segment σ^{k+1}. Hence, M will classify both f_1 and f_2 as in the same set, a contradiction. ☒

Moreover, it is also possible to prove a hierarchy in the number of mind changes that is not related to the number of sets to be classified.

Theorem 9. *For any $c \in \mathbb{N}$ there exist pairwise disjoint sets $S_0, S_1 \subseteq \mathcal{R}_{0,1}$ such that $(S_0, S_1) \in CL_2^{c+1} \setminus CL_2^c$.*

Proof. Let $c \in \mathbb{N}$ be arbitrarily fixed. Furthermore, let g be a $\{0,1\}$ valued recursive function satisfying

(1) for all $i \in \mathbb{N}$, $g(i,n) = 0$ for almost all $n \in \mathbb{N}$ or $g(i,n) = 1$ for almost all $n \in \mathbb{N}$,

(2) for all $i \in \mathbb{N}$, $card(\{n \mid n \in \mathbb{N},\ g(i,n) \neq g(i,n+1)\}) \leq c+1$,

(3) there is no recursive function h such that

(3.1) $h(i,n) = g(i,n)$ for all $i \in \mathbb{N}$ and almost all $n \in \mathbb{N}$,

(3.2) $card(\{n \mid n \in \mathbb{N},\ h(i,n) \neq h(i,n+1)\}) \leq c$ for all $i \in \mathbb{N}$.

In other words, g is a $\{0,1\}$ valued limiting recursive function that can be computed with at most $c+1$ mind changes but no limiting recursive function can compute the limits of g with at most c mind changes. The existence of such function g has been proved by Gold (1965).

Now we define $S_0, S_1 \subseteq \mathcal{R}_{0,1}$ as follows: $S_0 = \{0^i 1^\infty \mid i \in \mathbb{N}$ and $g(i,n) = 0$ for almost all $n\}$, and $S_0 = \{0^i 1^\infty \mid i \in \mathbb{N}$ and $g(i,n) = 1$ for almost all $n\}$.

By definition, $S_0 \cap S_1 = \emptyset$. A CM M witnessing $(S_0, S_1) \in CL_2^{c+1}$ first counts the number of consecutive zeros until it receives the first 1, and outputs nothing. Then, on any input $0^i 1^n$ it computes and outputs $g(i, i+n)$. In accordance with the choice of function g it directly follows that $(S_0, S_1) \in CL_2^{c+1}(M)$. On the other hand, it is easy to see that any CM $\hat{M}$ with $(S_0, S_1) \in CL_2^c(\hat{M})$ may be used to design a program for a recursive function h satisfying (3.1) and (3.2), a contradiction to the choice of g. Hence, the theorem follows. ☒

6. Classification versus Multi–Classification and Consistent Classification

In this section we compare the power of classification, multi–classification and consistent classification. In particular, we are interested in learning whether or not multi–classification or consistent classification is harder to achieve than ordinary one. Since neither multi–classification nor consistent classification has been studied in the framework presented here, our goal is twofold. First we are interested in general results concerning the classification power of these models. Second, we ask whether or not the results obtained extend to the classification of predicates.

Our next theorem compares multi–classification and ordinary classification in the finite case.

Theorem 10. *Multi–*$FCL_2 \subset FCL_2$

Proof. Let K be the halting set, i.e., $K = \{k \mid k \in \mathbb{N},\ \varphi_k(k)$ is defined$\}$. We define sets S_0, S_1 as follows. Let $S_0 = \{f \mid f \in \mathcal{R}$ and $[f(0) \in K$ or $(f(0), f(1) \in K)]\}$ and $S_1 = \{f \mid f \in \mathcal{R}$ and $[f(1) \in K$ or $(f(0), f(1) \in K)]\}$.

By definition, for any $f \in S_0 \cup S_1$ we have $f(0) \in K$ or $f(1) \in K$. Hence, a CM witnessing $(S_0, S_1) \in FCL$ may work as follows. After having received both $f(0)$ and $f(1)$ it tries to compute $\varphi_{f(0)}(f(0))$ and $\varphi_{f(1)}(f(1))$. One of these computations has to stop. In case the CM verifies $f(0) \in K$, it outputs "0." If the CM verifies $\varphi_{f(1)}(f(1))$ is defined, then it outputs "1." Clearly, $(S_0, S_1) \in FCL(M)$.

By way of contradiction assume that $(S_0, S_1) \in$ *Multi–FCL*(M) for some CM M. Then the halting set K is decidable by the following procedure.

On input $z \in \mathbb{N}$ the procedure chooses an arbitrary $k \in K$. Then it uses M to finitely multi–classify $f = kz0^\infty$. The procedure feeds successively $f(0), f(1), ...$ to M until M produces its hypothesis. If M outputs "0" then the procedure outputs "$z \notin K$." In case M's output is "$(0,1)$" the procedure outputs "$z \in K$."

It remains to show that the procedure defined above does work correctly. First, since $f(0) \in K$, the function f belongs to $S_0 \cup S_1$. Hence, M must finitely multi–classify f. Therefore, when fed $f(0), f(1), ...$ the CM M has to output a correct hypothesis. The rest directly follows from the definition of S_0 and S_1. ☒

Applying a simple coding technique, the proof of Theorem 10 directly yields the following corollary.

Corollary 11. *There are sets $S_0, S_1 \subseteq \mathcal{R}_{0,1}$ such that $(S_0, S_1) \in FCL_2 \setminus$ Multi–FCL_2.*

Next we aim to characterize multi–classification in terms of standard classification. For that purpose, we introduce the notion of a *mosaic*.

Let $S_0, ..., S_{k-1} \subseteq \mathcal{R}$ and $S = S_0 \cup \ ... \ \cup S_{k-1}$. Let $ALL = \{0, ..., k-1\}$ and $SUB \subseteq ALL$. Then let $S_{SUB} = \bigcap_{j \in SUB} S_j \setminus \bigcup_{i \in ALL \setminus SUB} S_i$. Clearly, S_{SUB} is the set of all $f \in S$ that belong to any of the classes S_j, $j \in SUB$, and to none of the classes S_i, $i \in ALL \setminus SUB$. Finally, let $mosaic(S_0, ..., S_{k-1})$ be the tuple of all sets S_{SUB}, where $SUB \subseteq ALL$.

Obviously, $mosaic(S_0, ..., S_{k-1})$ is a 2^k–tuple of pairwise disjoint sets the union of which is S (cf. Figure 1). Note that some components of $mosaic(S_0, ...S_{k-1})$ may be empty.

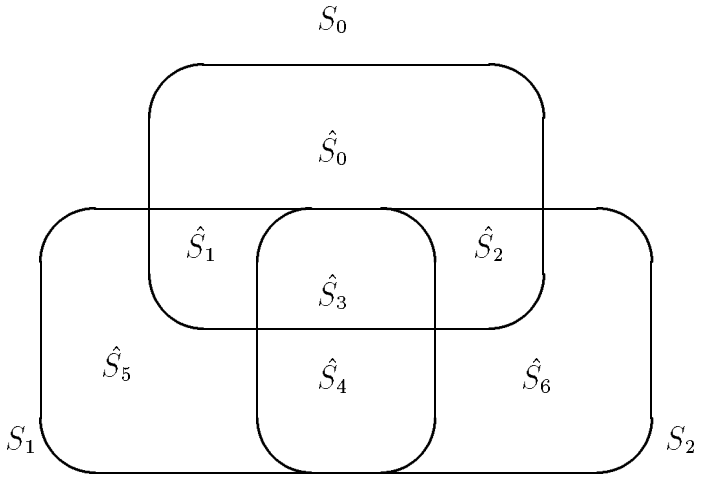

Figure 1: The mosaic $\hat{S}_0$, ..., $\hat{S}_6$, $\hat{S}_7$ of the sets S_0, S_1, S_2, where $\hat{S}_7 = \emptyset$.

Finally, let $mosaic(S_0, ..., S_{k-1}) = (T_0, ..., T_{2^k-1})$. Moreover, without loss of generality we assume that the T_j's, $0 \leq j \leq 2^k - 1$, are ordered in such a way that from the index j the corresponding set SUB can be computed such that $S_{SUB} = T_j$ and vice versa. Formally, this can be realized by a one-to-one mapping m from the set of all subsets SUB of ALL onto the set $\{0, ...2^k - 1\}$ such that for any $SUB \subseteq ALL$, $S_{SUB} = T_{m(SUB)}$.

We have obtained the following characterization.

Theorem 12. *For all $k \geq 2$ and all sets $S_0, ..., S_{k-1} \subseteq \mathcal{R}$ we have:*
$(S_0, ..., S_{k-1}) \in$ Multi–CL if and only if $mosaic(S_0, ..., S_{k-1}) \in CL$.

Proof. Necessity. Let $(S_0, ..., S_{k-1}) \in Multi\text{–}CL(M)$. Let $\hat{M}$ be a CM such that for any $f \in S_0 \cup \ ... \ \cup S_{k-1}$ and any $n \in \mathbb{N}$, $\hat{M}(f^n) = m(M(f^n))$. Clearly, $mosaic(S_0, ..., S_{k-1}) \in CL(\hat{M})$.

Sufficiency. Let $mosaic(S_0, ..., S_{k-1}) \in CL(M)$ for some CM M. Furthermore, let $\hat{M}$ be a CM such that for any $f \in S_0 \cup \ ... \ \cup S_{k-1}$ and any $n \in \mathbb{N}$, $\hat{M}(f^n) = m^{-1}(M(f^n))$. Obviously, $(S_0, ..., S_{k-1}) \in Multi\text{–}CL(\hat{M})$. ☒

Finally in this section we relate consistent classification to ordinary one.

Theorem 13. $Cons\text{–}CL \subset CL$

Proof. Let K be the halting set, i.e., $K = \{k \mid k \in \mathbb{N}, \ \varphi_k(k) \text{ is defined}\}$. First we define two disjoint sets S_0, S_1 splitting $\mathcal{R}$ such that $(S_0, S_1) \in CL_2^1 \setminus Cons\text{–}CL_2$. Let $S_0 = \{f \mid f \in \mathcal{R}, \ f(0) \in K\}$ and $S_1 = \{f \mid f \in \mathcal{R}, \ f(0) \notin K\}$. Clearly, $S_0 \cap S_1 = \emptyset$ and $S_0 \cup S_1 = \mathcal{R}$. An CM M witnessing $(S_0, S_1) \in CL_2^1(M)$ may work as follows. On any input f^n, $f \in \mathcal{R}$, $n \in \mathbb{N}$ the machine M tries to compute $\varphi_{f(0)}(f(0))$ within at most n steps. If the computation of $\varphi_{f(0)}(f(0))$ stops within this time bound then M outputs "0." Otherwise, it conjectures "1." Obviously, $(S_0, S_1) \in CL_2^1(M)$.

It remains to show that $(S_0, S_1) \notin Cons\text{–}CL$. Suppose the converse, i.e., there is a CM M that consistently classifies (S_0, S_1). As we shall show, the existence of such a CM implies the decidability of the halting set, a contradiction.

Claim. If $(S_0, S_1) \in Cons\text{–}CL(M)$ then any program for M can be used to construct effectively an algorithm that decides K.

The wanted algorithm $\mathcal{A}$ is defined as follows. On any input $z \in \mathbb{N}$ the algorithm $\mathcal{A}$ uses M to classify consistently $f = z^\infty$. More precisely, $\mathcal{A}$ simulates the computation of $M(f^0)$, $M(f^1)$, ... until M, when successively fed $f(0)$, $f(1)$, ... outputs its first guess j. If $j = 0$, then $\mathcal{A}$ outputs "$z \in K$." In case $j = 1$ $\mathcal{A}$ outputs "$z \notin K$."

Obviously, $\mathcal{A}$ is recursive. It remains to show that $\mathcal{A}$ works correctly. Suppose $j = 0$. In accordance with the definition of consistent classification there has to be a function $g \in S_0$ such that f and g do coincide up to n, where n is the least number such that M, when fed f^n outputs its first conjecture. In particular, $g(0) = f(0) = z$. Hence, $z \in K$ and $\mathcal{A}$ behaves correctly. Finally, suppose $j = 1$. Again there has to be a $g \in \mathcal{R}$ such that $g \in S_1$ and $g(0) = f(0) = z$. Hence, $z \notin K$.

This proves the claim and hence the theorem is demonstrated. ☒

Corollary 14. *There are sets* $S_0,\ S_1 \subseteq \mathcal{R}_{0,1}$ *such that* $(S_0, S_1) \in CL \setminus Cons\text{–}CL$.

Proof. Let K be as above. We set $S_0 = \{1^i 0^\infty \mid i \in K\}$ and $S_1 = \{1^i 0^\infty \mid i \notin K\}$. Then the same arguments as in the proof of Theorem 13 apply mutatis mutandis. We omit the details. ☒

7. Classification of Languages

In this section first we examine the classification of the regular languages (cf. Lewis and Papadimitriou (1981)). The regular languages can be modeled as predicates as well. This is done by fixing an isomorphism between strings over the alphabet of the regular language and the natural numbers. Then a (regular) language represented by a $\{0,1\}$ valued function f is the set of strings that correspond to the natural numbers in the set $\{x \mid f(x) = 1\}$. The details of this encoding will be suppressed as much as possible without sacrificing clarity. By a *positive example* we mean a string that corresponds to a value x such that $f(x) = 1$. Similarly, by a *negative example* we mean a string that corresponds to a value x such that $f(x) = 0$.

Theorem 15. *It is impossible to classify an arbitrary language from positive and negative examples as being either regular or not regular.*

Proof. As a first step, we restate the theorem more formally. Let S_0 be the class of regular languages, i.e., the set of $\{0,1\}$ valued functions that correspond to the regular languages. Let S_1 be all the other languages, e.g. the rest of the $\{0,1\}$ valued functions. We prove that $(S_0, S_1) \notin CL_2$. Suppose by way of contradiction that M is a CM such that $(S_0, S_1) \in CL_2(M)$.

We construct a language L over the alphabet $\{a, b\}$, by specifying positive and negative examples. For the sake of presentation we omit the corresponding encoding into a $\{0,1\}$ valued function. Once a string is given as a positive (negative) example, it can never late be given as a negative (positive) example. The *least next example* is the example corresponding to the lexicographically least string that has not yet be presented as either a positive or negative example.

The specification of L is initiated by declaring more and more least next examples as negative until M, given the examples so far, produces a conjectured classification. If M fails to produce a conjecture, then this initialization step will, ultimately, give *every* string as a negative example. In this case, L is the empty language, which is regular. Furthermore, M will fail to correctly classify L.

Let $i \in \{0,1\}$ be the most recent conjecture produced by M after seeing all the examples that have been determined so far. Based on the value of i, new examples are added to L, until (if ever) enough data is added to entice M to produce a new conjecture.

If $i = 0$: Let x be the least next example. If x corresponds to a string of the form $a^n b^n$, for some n, then add this example as a positive example. Otherwise, the example becomes a negative one.

If $i = 1$: Add the least next example as a negative example.

If the above procedure is successful in always convincing M to change its conjectured classification, then M fails to classify L as either a regular language or not. If M does eventually settle on a classification for L, then one of the two cases above will add the the rest of the examples to L. In the first case, when M determines that L is regular, L will end up being a finite variant of the language $\{a^n b^n \mid n \in \mathbb{N}\}$ which is not regular. Since the regular languages are closed under finite variations, L cannot be regular. Hence, M fails to correctly classify L. In the second case, when M determines that L is not regular, L will end up being a finite language. Hence, again, M fails to correctly classify L. ☒

By way of contrast with Theorem 15, it is possible to separate arbitrarily large subsets of the regular languages ¿from the rest.

Theorem 16. *Let n be an arbitrary natural number. Let S_0 be the set of all regular languages that are recognized by some k-state finite automaton, $k \leq n$. Let S_1 be all the languages not in S_0. Then $(S_0, S_1) \in CL$.*

Proof. There are only finitely many k state finite automata, $k \leq n$, over any given alphabet. A CM that classifies (S_0, S_1) initially conjectures "0" and only changes to "1" when it observes that no automaton in the finite set of n-state automata is consistent with the examples observed as input. ☒

Next we ask whether or not Theorem 16 may be generalized. Disregarding the obvious direction to handle more complex cases of finite sets of languages we consider the problem under what circumstances infinite sets of languages are separable from the rest. As we have already seen, this is not always the case. Moreover, Theorem 15 allows a conclusion that is interesting in its own right. On the one hand, the set of all regular languages is even reliably learnable on the set of all total functions (cf. Blum and Blum (1975)). On the other hand, no algorithm that learns the set of all regular languages can be converted into a classification machine as Theorem 15 shows. Hence, it is worth to ask for an explanation of that phenomenon.

Analyzing the regular languages ¿from an algorithmic point of view yields that they possess several favorable properties. First, the set of all regular languages is recursively enumerable. Second, membership in the regular languages is uniformly decidable. Finally, the regular languages possess several structural properties that may be tested algorithmically. Nevertheless, they are not separable, even in the limit, from the rest of all languages. This yields the conjecture that their unclassifiability might be caused not by their algorithmic but their topological properties. Namely, ¿from a topological point of view the set of all regular languages is dense, i.e., it

consists only of accumulation points. As a consequence, there is no learning algorithm inferring it that does not exceed any a priorily fixed number of allowed mind changes. On the other hand, for any fixed n, the set of all regular languages acceptable by some n-state finite automaton may be obviously inferred within an a priorily fixed number of mind changes.

Therefore, we try to generalize Theorem 16 in the direction that any learning algorithm not exceeding a number of mind changes fixed in advance may be converted into a classification machine.

For that purpose we need some notation. Let Σ be any fixed alphabet. A recursively enumerable set $\mathcal{L} = (L_j)_{j \in \mathbb{N}}$ of languages over Σ is said to be an indexed family, if all languages L_j are non–empty, and there is an algorithm that uniformly decides membership in L_j for all $j \in \mathbb{N}$ and all strings $s \in \Sigma^*$ (cf. Angluin (1980)). That means $\mathcal{L} = (L_j)_{j \in \mathbb{N}}$ is an indexed family iff there is a recursive function f such that for all numbers j and all strings $s \in \Sigma^*$ we have

$$f(j,s) = \begin{cases} 1, & \text{if} \quad s \in L_j, \\ 0, & \text{otherwise.} \end{cases}$$

Assuming Σ^* to be lexicographically ordered, we may describe $\mathcal{L}$ by a sequence $\mathcal{F} = (f_j)_{j \in \mathbb{N}}$ of uniformly recursive predicates, where $f_j(s) = f(j,s)$ for all $j \in \mathbb{N}$ and all $s \in \Sigma^*$. Moreover, in the sequel we identify 2^{Σ^*} with the set of all $\{0,1\}$ valued total functions.

The inferability of indexed families has attracted a lot of attention in learning theory (cf. Lange and Zeugmann (1992a), (1993) and the references therein). As it turned out, when dealing with the inferability of indexed families the choice of the hypothesis space is of importance. It may affect both, the learnability at all as well as the efficiency (cf. Lange and Zeugmann (1992a), (1993)). In particular, not requiring an IIM to learn within the given enumeration $\mathcal{F} = (f_j)_{j \in \mathbb{N}}$ may reduce the number of necessary mind changes. Therefore, in all what follows we consider *class preserving* learning, i.e., when inferring an indexed family $\mathcal{F} = (f_j)_{j \in \mathbb{N}}$ we allow any suitable chosen hypothesis space $\mathcal{G} = (g_j)_{j \in \mathbb{N}}$ of uniformly recursive predicates such that any predicate in $\mathcal{F}$ possesses a description in $\mathcal{G}$ and any hypothesis g_j describes a language from $\mathcal{F}$, i.e., $range(\mathcal{G}) = range(\mathcal{F})$.

Now we are ready to present the next theorem.

Theorem 17. *Let $m \in \mathbb{N}$, $m \geq 1$, and let $\mathcal{F}$ be any indexed family over some fixed alphabet Σ that can be learned from positive and negative data with at most m mind changes. Then there exists a partition of 2^{Σ^*} into $m+2$ pairwise disjoint classes $\mathcal{F}_0, ..., \mathcal{F}_{m+1}$ such that*

(1) $\mathcal{F}_0 \cup ... \cup \mathcal{F}_m = \mathcal{F}$,

(2) $(\mathcal{F}_0, ..., \mathcal{F}_{m+1}) \in CL$.

Proof. Let $\mathcal{F} \in EX_m(\hat{M})$ w.r.t. a hypothesis space $\hat{\mathcal{G}}$. It is very suggestive to define the wanted partition to be the sets of all $\{0,1\}$ valued functions that are inferred by $\hat{M}$ with exactly i mind changes, for $0 \leq i \leq m$ and $\mathcal{F}_{m+1}$ to be the rest. However, before doing something similar there are some problems we have to overcome in advance. The main difficulty we have to deal with is that we do not know how $\hat{M}$ behaves on inputs f that do not belong to $\mathcal{F}$. In particular, since we want to partition the set of all total $\{0,1\}$ valued function it might even be that $\hat{M}$ is not defined on these inputs. But even if it is, it might converge on functions not belonging to $\mathcal{F}$ after having performed $c \leq m$ mind changes. We overcome these difficulties by applying the characterization theorem of Lange and Zeugmann (1992a) (cf. Theorem 12). Applying this theorem we obtain a class preserving hypothesis space $\mathcal{G} = (g_j)$ and a total and consistently working IIM M such that $\mathcal{F} \in EX_m(M)$ w.r.t. $\mathcal{G}$. That means, on any input f^n, if M outputs a hypothesis j, then $f(x) = g(x)$ for all $x \leq n$. Moreover, membership in $\mathcal{G}$ is uniformly decidable. Now we can define the desired partition as follows. For $0 \leq i \leq m$ we set $\mathcal{F}_i = \{f \mid f$ can be learned by M with exactly i mind changes $\}$, and define $\mathcal{F}_{m+1} = 2^{\Sigma^*} \setminus \bigcup_{0 \leq i \leq m} \mathcal{F}_i$.

It remains to prove that all stated properties are fulfilled.

Obviously, $\mathcal{F}_0, ..., \mathcal{F}_{m+1}$ is a partition of 2^{Σ^*}. For the purpose to prove (1) we first recall that any $f \in \mathcal{F}$ is learned by M with at most m mind changes. Hence, $\mathcal{F} \subseteq \mathcal{F}_0 \cup ... \cup \mathcal{F}_m$. For the opposite direction suppose an $f \notin \mathcal{F}$ but $f \in \mathcal{F}_i$ for some $i \in \{0, ..., m\}$. Consequently, M performs on input $f(0), f(1), ...$ exactly i mind changes and then it always outputs a hypothesis j. Let $g = g_j$. Taking into account that M works consistently, we may conclude that M verifies $f(x) = g(x)$ for all $x \in \mathbb{N}$. Hence, $f = g$, and therefore $f \in \mathcal{F}$ since $range(\mathcal{F}) = range(\mathcal{G})$. This contradiction proves (1).

It remains to define a CM $\tilde{M}$ such that $(\mathcal{F}_0, ..., \mathcal{F}_{m+1}) \in CL(\tilde{M})$. Let $f \in 2^{\Sigma^*}$ and $n \in \mathbb{N}$. We set

$\tilde{M}(f^n) =$ "Simulate $\hat{M}$ when successively fed $f(0), ..., f(n)$. If $\hat{M}$ outputs a hypothesis and has performed exactly i, $0 \leq i \leq m$ mind changes after having read $f(0), ..., f(n)$, then output "i," and request the next input.
Otherwise, output "$m+1$," and request the next input."

Arguing as above one easily proves that $\tilde{M}$ classifies $\mathcal{F}_0, ..., \mathcal{F}_{m+1}$. ☒

The latter theorem directly allows the following corollary.

Corollary 18. *Let $m \geq 1$, and let $\mathcal{F}$ be any indexed family over some fixed alphabet Σ that can be learned from positive and negative data with at most m mind changes. Then it is possible to classify an arbitrary language from positive and negative data as belonging to $\mathcal{F}$ or not.*

Looking at all the results obtained above we see that we have always dealt with complete information concerning the objects to be learned or classified. However, a

huge part of language learning theory is devoted to learning from positive data only. Hence, it is only natural to consider classification of languages from positive data. What we like to present at the end of this section is a short outlook into this setting.

Considering learning or classification from positive data requires some carefulness. First of all, we have to deal with the order of information presentation. Clearly, we cannot assume to receive the data in lexicographical order, since this would implicitly deliver much more information than allowed. Consequently, one demands a CM or IIM to learn on *all* sequences of positive data that eventually contain every string from the language under consideration. More precisely, let L be a language and $t = s_0, s_1, s_2, ...$ an infinite sequence of strings from Σ^* such that $range(t) = \{s_k \mid k \in \mathbb{N}\} = L$. Then t is said to be a *positive presentation* for L or, synonymously, a *text*. We define $Text(L)$ to be the set of all texts for L. Moreover, let t be a text and let $x \in \mathbb{N}$: then t_x denotes the initial segment of t of length $x + 1$. Note that we do not require a text to be computable. Now we ask whether or not we may extend Theorem 17 to the case of learning from positive data. The answer is twofold. It is still possible to transform a learning algorithm that works with a number of mind changes fixed a priorily into a classification machine. Nevertheless, we conjecture that it is no longer possible to partition 2^{Σ^*}. However, we still get a partition of the indexed family. For the sake of readability we present the next theorem in terms of languages.

Theorem 19. *Let $m \in \mathbb{N}$ and let $\mathcal{L}$ be any indexed family over some fixed alphabet that can be learned from positive data with at most m mind changes. Then there exist pairwise disjoint classes $\mathcal{L}_0, ..., \mathcal{L}_m$ such that*

(1) $\mathcal{L}_0 \cup ... \cup \mathcal{L}_m = \mathcal{L}$,

(2) $(\mathcal{L}_0, ..., \mathcal{L}_m) \in CL$.

Proof. Before defining the desired partition we introduce the notion of canonical text that will be very helpful in proving the theorem.

Let $\mathcal{L}$ be an indexed family and let L any of its languages. Moreover, let $s_1, s_2, ...$ be the lexicographically ordered text of Σ^*. The canonical text t of L is obtained as follows. Test sequentially whether $s_z \in L$ for $z = 1, 2, 3, ...$ until the first z is found such that $s_z \in L$. Since $L \neq \emptyset$ there must be at least one z fulfilling the test. Set $t_1 = s_z$. We proceed inductively, $x \geq 1$:

$$t_{x+1} = \begin{cases} t_x s_{z+x}, & \text{if} \quad s_{z+x} \in L \\ t_x s, & \text{otherwise, where } s \text{ is the last string in } t_x \end{cases}$$

By assumption, there is an IIM M and a class preserving space of hypothesis $\mathcal{G} = (G_j)_{j \in \mathbb{N}}$ such that M learns any $L \in \mathcal{L}$ from any text $t \in Text(L)$ with at most m mind changes. Now we can define the wanted partition of $\mathcal{L}$ as follows: For $0 \leq i \leq m$ we set $\mathcal{L}_i = \{L | L \in \mathcal{L},\ L$ can be learned by M ¿from its canonical text with exactly i mind changes $\}$. Obviously, (1) is fulfilled. It remains to define a CM

$\tilde{M}$ that classifies $(\mathcal{L}_0, ..., \mathcal{L}_m)$. The main problem we have to deal with is the following. The CM $\tilde{M}$ has to classify every $L \in \mathcal{L}$ from all texts $t \in \mathit{Text}(L)$. Therefore, we cannot simply simulate M and count the number of mind changes, since this number may depend on the particular text the IIM is fed. Hence, our goal must be to simulate M on the canonical text of the language we have to classify. However, for that purpose we should know at least one of L's indices in $\mathcal{G}$.

But such an index will be constructed in the limit by the IIM M just on any text $t \in \mathit{Text}(L)$. Hence the desired CM $\tilde{M}$ works on arbitrary text $t \in \mathit{Text}(L)$ as follows: It simulates M on t_x, $x \in \mathbb{N}$, until M will eventually converge. For any hypothesis j produced by M on t, and hence also for the final and correct hypothesis, $\tilde{M}$ then simulates M on the initial segment t^j_x of the *canonical* text t^j of $L(G_j)$, counts the number of mind changes and outputs it as its actual hypothesis for classification.

Now we formally define the CM $\tilde{M}$. Let $L \in \mathcal{L}$, $t \in \mathit{Text}(L)$ and $x \in \mathbb{N}$. We set:

$\tilde{M}(t_x) =$ "Compute $j = M(t_x)$.
Compute the number z of mind changes of M when successively fed t^j_x.
If $z \leq m$, then output z and request the next input.
If $z > m$ output nothing and request the next input."

It remains to show that $\tilde{M}$ correctly classifies all the languages ¿from $\mathcal{L}$. Let $L \in \mathcal{L}$, and $0 \leq i \leq m$ be such that L can be learned by M from its canonical text with exactly i mind changes, i.e., $L \in \mathcal{L}_i$. Furthermore, let x be such that both

- $j = M(t_{x+r})$ for all $r \in \mathbb{N}$ (hence $L = L(G_j)$),
- on t^j_x the machine M performs exactly i mind changes.

is fulfilled. Obviously, $\tilde{M}(t_y) = i$ for all $y \geq x$. Hence, $(\mathcal{L}_0, ..., \mathcal{L}_m) \in CL(\tilde{M})$. ☒

8. Conclusions and Open Problems

We have studied the classification of $\{0,1\}$ valued recursive functions, and simultaneously the classification of languages from positive and negative data. Moreover, extending previous work by Wiehagen and Smith (1992) we introduced new models of classification, i.e., classification with a bounded number of mind changes, multi-classification and consistent classification. We related all these classification types one to the other, thereby showing what they have in common and where the differences are. However, there are several question that deserve further study.

First, it would be desirable to gain a deeper understanding under what circumstances the restriction of arbitrary recursive function classes to predicates is classifiable, provided the original function classes are not. Second, the impact of consistent

classification should be investigated in some more detail. Looking at potential applications there are several scenarios where consistent classification is preferable. Nevertheless, as we have seen, this requirement might prevent one at all in successfully designing a classification machine. Hence, it seems to be highly desirable to elaborate sufficient and necessary conditions for consistent classification. Third, we would like to suggest to deal with the characterization of all the classification types introduced. Characterizations play an important role in learning theory (cf. e.g. Blum and Blum (1975), Angluin (1980), Zeugmann (1983), Wiehagen (1991), Lange and Zeugmann (1992b)). As it turned out, most of the characterizations obtained lead to a better understanding into the problem how algorithms performing the learning process may be designed. Hence, characterizing classification might yield a deeper insight into the nature of classification. Moreover, this might help to gain a better understanding of the complex relation between classification and learning.

Finally, we have mainly dealt with the classification of languages ¿from positive and negative data. Nevertheless, from the point of view of potential applications classification from positive data deserves attention as well. We regard the last theorem in the latter section as a starting point for further research in that direction. In this context, Fulk's (1990) results suggest interesting problems. In particular, Fulk (1990) studied the impact of several demands on the learning power of IIMs, e.g.; prudence, rearrangement independence or set–driveness. Since these requirements reflect postulates of naturalness, it is worth to deal with their influence on the power of CMs.

9. References

ANGLUIN, D. (1980), Inductive inference of formal languages from positive data, *Information and Control* **45**, 117 - 135.

ANGLUIN, D., AND SMITH, C.H. (1983), Inductive inference: theory and methods, *Computing Surveys* **15**, 237 - 269.

ANGLUIN, D., AND SMITH, C.H. (1987), Formal inductive inference, *in* "Encyclopedia of Artificial Intelligence" (St.C. Shapiro, Ed.), Vol. 1, pp. 409 - 418, Wiley-Interscience Publication, New York.

BARZDIN. J. (1971), Complexity and frequency solution of some algorithmically unsolvable problems, Doct. Diss., Novosibirsk, State University, (in Russian).

BLUM, L., AND BLUM, M. (1975), Toward a mathematical theory of inductive inference, *Information and Control* **28**, 122 - 155.

CASE, J., AND SMITH, C.H. (1983), Comparison of identification criteria for machine inductive inference, *Theoretical Computer Science* **25**, 193 - 220.

DUDA, R., AND HART, P. (1973), "Pattern Classification and Scene Analysis," Wiley-Interscience Publication, New York.

FREIVALDS, R., KINBER, E.B., AND WIEHAGEN, R. (1992), Convergently versus divergently incorrect hypotheses in inductive inference, GOSLER Report 05/92, January 1992, Fachbereich Mathematik und Informatik, TH Leipzig.

FULK, M. (1990), Prudence and other restrictions in formal language learning, *Information and Computation* **85**, 1 - 11.

FREIVALDS, R.V., AND WIEHAGEN, R. (1979), Inductive inference with additional information, *Journal of Information Processing and Cybernetics (EIK)* **15**, 179 - 184.

GOLD, M.E. (1965), Limiting recursion, *Journal of Symbolic Logic* **30**, 28 - 48.

GOLD, M.E. (1967), Language identification in the limit, *Information and Control* **10**, 447 - 474.

JANTKE, K.P., AND BEICK, H.R. (1981). Combining postulates of naturalness in inductive inference, *Journal of Information Processing and Cybernetics (EIK)* **17**, 465 - 484.

LANGE, S., AND ZEUGMANN, T. (1992a), Learning recursive languages with bounded mind changes, GOSLER–Report 16/92, FB Mathematik und Informatik, TH Leipzig, September 1992.

LANGE, S., AND ZEUGMANN, T. (1992b), Types of monotonic language learning and their characterization, *in* "Proceedings 5th Annual ACM Workshop on Computational Learning Theory," Pittsburgh, pp. 377 - 390, ACM Press.

LANGE, S., AND ZEUGMANN, T. (1993), Language learning in dependence on the space of hypotheses, *in* "Proceedings 6th Annual ACM Conference on Computational Learning Theory," Santa Cruz, pp. 127 - 136, ACM Press.

LEWIS, H., AND PAPADIMITRIOU, C. (1981), "Elements of the Theory of Computation," Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

MACHTEY, M., AND YOUNG, P. (1978), "An Introduction to the General Theory of Algorithms," North-Holland, New York.

MICHALSKI, R.S., CARBONELL, J.G., AND MITCHELL, T.M. (1983), "Machine Learning," Tioga Publishing Co., Palo Alto, CA.

OSHERSON, D., STOB, M., AND WEINSTEIN, S. (1986), "Systems that Learn, An Introduction to Learning Theory for Cognitive and Computer Scientists," MIT-Press, Cambridge, Massachusetts.

WIEHAGEN, R. (1991), A thesis in inductive inference, *in* "Proceedings First International Workshop on Nonmonotonic and Inductive Logic," Karlsruhe, December 1990, J.Dix, K.P. Jantke and P.H. Schmitt (Eds.), Lecture Notes in Artificial Intelligence 543, pp. 184 - 207, Springer-Verlag.

WIEHAGEN, R., AND SMITH, C. (1992), Classification versus generalization, *in* "Proceedings 5th Annual ACM Workshop on Computational Learning Theory," Pittsburgh, pp. 224 - 230, ACM Press, New York.

ZEUGMANN, T. (1983), A-posteriori characterizations in inductive inference of recursive functions, *Journal of Information Processing and Cybernetics (EIK)* **19**, 559 - 594.

ZEUGMANN, T. (1988), On the power of recursive optimizers, *Theoretical Computer Science* **62**, 289 - 310.