Learning by Switching Type of Information [*]

Sanjay Jain
School of Computing
National University of Singapore
3 Science Drive 2
Singapore 117543
Email: sanjay@comp.nus.edu.sg

Frank Stephan
Mathematisches Institut
Im Neuenheimer Feld 294
Universität Heidelberg
69120 Heidelberg, Germany, EU
Email: fstephan@math.uni-heidelberg.de

Abstract
The present work is dedicated to the study of modes of data-presentation in the range between text and informant within the framework of inductive inference. In this study, the learner alternatingly requests sequences of positive and negative data. We define various formalizations of valid data presentations in such a scenario. We resolve the relationships between these different formalizations, and show that one of these is equivalent to learning from informant. We also show a hierarchy formed (for each of the formalizations studied) by considering the number of switches between requests for positive and negative data.

1 Introduction

Astronomers observing the sky with telescopes cannot obtain all available information but have to focus their study on selected areas and might from time to time change to another area of the sky. Forty years after the discovery of Uranus, it was found that Uranus was not following the predicted orbit exactly. Taking into account the influence of the other known planets, the astronomer Alexis Bouvard came up with the hypothesis that there exists a further unknown planet which disturbs the orbit of Uranus. John Couch Adams and Urbain Jean Joseph Le Verrier both computed independently the position of the unknown planet. In 1846, Le Verrier communicated his results to Johann Gottfried Galle, who then found Neptune with his telescope at the given position.

Similar to astronomy, one can also in inductive inference consider the scenario that the learner cannot track all available data but has to focus on some type of data and

[*]A preliminary version of this paper appeared in ALT 2001. We would like to thank the anonymous referees for useful comments. Sanjay Jain was supported in part by NUS grant number R252-000-127-112. Frank Stephan was supported by the Deutsche Forschungsgemeinschaft (DFG) under the Heisenberg grant Ste 967/1–1.

can only few times switch the focus of attention. The purpose of the present work is to formalize such switching between the two main modes of data-presentation in inductive inference, namely between reading positive data which are elements of the set to be learned or negative data which are the non-elements. There are several ways to formalize this and it is investigated how these formalizations relate to each other and how they fit into the hierarchy of the already established notions of learning from positive data (text) or both, positive and negative data (informant).

In the scenario of learning from positive data, the learner is fed all the elements and no non-elements of a language L (the so called *text* of L), in any order, at most one element at a time. The learner, as it is receiving the data, outputs a sequence of grammars. The learner is said to identify (learn, infer) L just in case the sequence of grammars converges to a grammar for L. A class of languages is learnable if some machine learns each language in the class. This is essentially the paradigm of identification in the limit (called **TxtEx**) introduced by Gold [11]. Gold also considered the situation of learning from informant, where the learner receives both positive and negative data, that is elements of the graph of the characteristic function of L (called *informant* for L) as input. This leads to the identification criterion known as **InfEx**.

Gold [11] showed a central result that learning from text is much more restrictive than learning from informant. Gold gave an easy example of a class which can be learned from informant but not from text: the collection consisting of one infinite recursively enumerable set together with all its finite subsets.

The main motivation for this work is to explore the gap between these two extreme forms of data-presentation. Previous authors have already proposed several methods to investigate this gap, some of these are described below.

Gasarch and Pleszkoch [10] considered allowing learners access to non-recursive oracles. However Jain and Sharma [14] showed that even the most powerful oracles do not permit to learn all recursively enumerable (or even all recursive) sets from texts whereas the oracle K allows one to learn all recursively enumerable sets from informant.

Restrictions on the texts (such as allowing only primitive recursive texts or ascending texts) reduce their non-regularity and permit to pass on further information implicitly [20, 23]. For example, ascending texts permit to reconstruct the complete negative information in the case of infinite sets, but fail to do so in the case of finite sets. Thus the class of one infinite set and all its finite subsets is still not learnable from ascending text. On the other hand, the class of all recursively enumerable languages can be learned from primitive recursive texts. Merkle and Stephan [18] also considered strengthening the text by permitting additional queries to retrieve information not contained in standard texts.

Motoki [19] and later Baliga, Case and Jain [2] added to the positive information of the text some, but not all, negative information about the language to be learned. They considered two notions of supplying the negative data: (a) there is a finite set of negative information $S \subseteq \overline{L}$ such that the learner always succeeds learning the language L from input S plus a text for L, and (b) there is a finite set $S \subseteq \overline{L}$ such that the learner always succeeds learning the language L from a text for L plus a text for a set H disjoint to L which contains S, that is, which satisfies $S \subseteq H \subseteq \overline{L}$. In case (a) one is able learn the class of all recursively enumerable languages. Thus, the notion (b) is the more interesting one.

The present work treats positive and negative data symmetrically and several of our notions are much less powerful than those notions considered by Baliga, Case and Jain [2]. The most convenient way to define these notions is to use the idea of a minimum adequate teacher as, for example, described by Angluin [1]. A learner requests positive or negative data-items from a teacher which has – depending on the exact formalization – to fulfill certain requirements. These formalizations (and also the number of switches permitted) then define the model. We consider three formalizations (called **BasicSwEx**, **RestartSwEx**, **NewSwEx**) of requirements a teacher needs to satisfy. The naturalness of this approach is witnessed by the fact that all classes separating the various formalizations can be defined in easy topological terms. Due to the topological nature of the separating classes, these results hold even if the learners are non-computable. Out of the three formalizations, **NewSwEx** turns out to be the most natural definition in the gap between **TxtEx**-learning and learning from informant. **RestartSwEx** (without constraints on number of switches) coincides with learning from informant, whereas **BasicSwEx** has some strange properties.

2 Preliminaries

Notation. Any unexplained recursion theoretic notation can be found in Roger's textbook [21]. The symbol $\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. Symbols $\emptyset$, $\subseteq$, $\subset$, $\supseteq$, and $\supset$ denote empty set, subset, proper subset, superset, and proper superset, respectively. Cardinality of a set S is denoted by $\text{card}(S)$. Domain and range of a partial function ψ is denoted by $\text{domain}(\psi)$ and $\text{range}(\psi)$, respectively.

Infinite sequences are mappings from $\mathbb{N}$ to $\mathbb{N} \cup \{\#\}$; finite sequences are mapping from $\{y \in \mathbb{N} : y < x\}$ (for some $x \in \mathbb{N}$) to $\mathbb{N} \cup \{\#\}$. In the first case, the length of the sequence is ∞, whereas in the second case its length is x. We denote the length of a sequence η by $|\eta|$. Sequences may take a special value # to indicate a pause (when considered as a source of data). Therefore the notion *content* is introduced to denote the set of the numbers contained within the range of some finite or infinite sequence. The content of a sequence η is defined as $\text{content}(\eta) = \text{range}(\eta) \cap \mathbb{N}$. Furthermore, if $x \leq |\eta|$, then $\eta[x]$ denotes the restriction of η to the domain $\{y \in \mathbb{N} : y < x\}$. We let σ and τ range over finite sequences. We denote the sequence formed by the concatenation of τ at the end of σ by $\sigma\tau$. Furthermore, we use σx to denote the concatenation of sequence σ and the sequence of length 1, which contains the element x.

By φ we denote a fixed *acceptable* programming system for the partial computable functions that are mapping $\mathbb{N}$ to $\mathbb{N}$ [17, 21]. By φ_i we denote the partial recursive function computed by the program with number i in the φ-system. Such a program i is a (characteristic) index for a set L if

$$\varphi_i(x) = \begin{cases} 1, & \text{if } x \in L; \\ 0, & \text{otherwise.} \end{cases}$$

Programs for enumeration procedures (so called r.e. indices) are not considered in the present work. From now on, we call the recursive subsets of $\mathbb{N}$ just languages and only consider characteristic indices and not enumeration procedures. The symbols L, H range over languages. $\overline{L}$ denotes the complement, $\mathbb{N} - L$, of L. The symbol $\mathcal{L}$ ranges over classes of languages.

Learning theory often also considers learning non-recursive but still recursively enumerable sets. In this work we restrict ourselves to the recursive case since, for notions of learning considered in this paper: (I) all inclusions hold for the case of recursive sets iff they hold for the case of recursively enumerable sets; (II) recursive sets already permit us to construct candidates for separations of learning criteria – our diagonalization proofs use mainly the topological properties. Furthermore, recursive sets have, compared to recursively enumerable sets, the advantage that their complement also possesses a recursive enumeration. This is an interesting property to have, as we are considering positive and negative information in a symmetric way.

Notation from Learning Theory. The main scenario of inductive inference is that a learner reads more and more data on an object and outputs a sequence of hypotheses which eventually converge to the object to be learned.

Definition 2.1 (Gold [11]) A *text* T for a language L is an infinite sequence such that its content is L, that is, T contains all elements of L but none of $\overline{L}$. $T[n]$ denotes the finite initial sequence of T with length n.

Definition 2.2 (Gold [11]) A *learner* (or learning machine) is an algorithmic device which computes a mapping from finite sequences into $\mathbb{N}$.

We let T range over texts and $\mathbf{M}$ range over learning machines. $\mathbf{M}(T[n])$ is interpreted as $\mathbf{M}$'s conjecture for the input language based on data $T[n]$. We say that $\mathbf{M}$ converges on T to i, (written $\mathbf{M}(T)\downarrow = i$) iff $(\forall^\infty n)\ [\mathbf{M}(T[n]) = i]$.

There are several criteria for a learning machine to be successful on a language. Below we define learning in the limit introduced by [11].

Definition 2.3 (Gold [11]) (a) $\mathbf{M}$ **TxtEx**-learns a language L from text T iff, for some index i for L, for almost all n, $\mathbf{M}(T[n]) = i$.
(b) $\mathbf{M}$ **TxtEx**-learns a language L (written: $L \in \mathbf{TxtEx}(\mathbf{M})$) just in case $\mathbf{M}$ **TxtEx**-learns L from each text for L.
(c) $\mathbf{M}$ **TxtEx**-learns a class $\mathcal{L}$ of languages (written: $\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})$) just in case $\mathbf{M}$ **TxtEx**-learns each language from $\mathcal{L}$.
(d) $\mathbf{TxtEx} = \{\mathcal{L} :$ some learner $\mathbf{M}$ **TxtEx**-learns $\mathcal{L}\}$.

The following propositions on learning from text are useful in proving some of our results.

Proposition 2.4 (Based on Proposition 2.2A by Osherson, Stob and Weinstein [20]) *Let L be any infinite language and Pos be a finite subset of L. Then $\{H : Pos \subseteq H \subseteq L \wedge card(L - H) \leq 1\} \notin$* **TxtEx**.

Proposition 2.5 (Gold [11]) *Let L be any infinite language. If $\mathcal{L}$ contains L and the sets $L \cap \{0, 1, \ldots, n\}$, for infinitely many $n \in \mathbb{N}$, then $\mathcal{L} \notin$* **TxtEx**.

We now generalize the concept of learning and permit the learners to request explicitly positive or negative data from a teacher in order to define learning by switching between types of information received.

Definition 2.6 Learning is a game between a learner $\mathbf{M}$ and a teacher T. Both send alternately information in the following way: in the k-th round (for ease of notation we start with round 0), the learner first sends a request $r_k \in \{+, -\}$; the teacher then answers with an information x_k; thereafter the learner outputs a hypothesis e_k. There are three types of interactive protocols between the learner and the teacher; every teacher satisfying the protocol is permitted.

(a) *The basic switch-protocol.* The teacher has two texts T_+ and T_- of L and $\overline{L}$, respectively. After receiving r_k the teacher transmits $T_{r_k}(k)$.

(b) *The restarting switch-protocol.* The teacher has two texts T_+ and T_- of L and $\overline{L}$, respectively. After receiving r_k the teacher computes the current position $l = \text{card}\{h : 0 \leq h < k \wedge r_h = r_k\}$ and transmits $T_{r_k}(l)$.

Intuitively, in restarting switch-protocol, one may consider learner as asking the "next item" from the selected text (of language or its complement).

(c) *The newtext switch-protocol.* The teacher sends an $x_k \in L \cup \{\#\}$, if $r_k = +$ and $x_k \in \overline{L} \cup \{\#\}$, if $r_k = -$. Furthermore, if there is a k such that $r_h = r_k$, for all $h \geq k$, and either $k = 0$ or $r_{k-1} \neq r_k$, then the sequence $x_k, x_{k+1}, \ldots$ is a text for L (if $r_k = +$) or a text for $\overline{L}$ (if $r_k = -$).

Intuitively, in newtext switch-protocol, the teacher starts with a new text for L or $\overline{L}$ every time a switch occurs.

A class $\mathcal{L}$ is learnable according to the given protocol iff there is a learner $\mathbf{M}$ such that, for every $L \in \mathcal{L}$ and for every teacher satisfying the protocol for this L, the hypotheses of the learner $\mathbf{M}$ converge to an index e of L. The corresponding learning-criteria are denoted by **BasicSwEx**, **RestartSwEx** and **NewSwEx**, respectively.

Note that $\mathbf{M}$ is a **TxtEx**-learner iff M always requests positive data ($r_k = +$ for all k). Therefore, all three notions are generalizations of **TxtEx**-learning.

In the following we define similar restrictions on the number of switches as has been done for the number of mind changes by Case and Smith [7] and Freivalds and Smith [8]. We consider counting number of switches by ordinals. The learner has a counter for an ordinal, which is downcounted at every switch. Due to the well-ordering of the ordinals, the counter can be downcounted only finitely often. In order to ensure that the learner is computable, we consider throughout this work only recursive ordinals. In particular, we use a fixed notation system, Ords, and a partial ordering of ordinal notations [16, 21, 22]. We use $\preceq, \prec, \succeq$ and $\succ$ to compare ordinals according to the partial ordering mentioned above. We do not go into the details of the notation system used, but instead refer the reader to the methods outlined in the papers [5, 8, 15, 16, 21, 22].

Definition 2.7 **BasicSw$_*$Ex** denotes the variant of **BasicSwEx** where the requests of **M** have to converge to some r, whenever **M** deals with a teacher following basic switch-protocol, for any given $L \in \mathcal{L}$.

For an ordinal notation α, we now define the variant **BasicSw$_\alpha$Ex** of **BasicSwEx**. The learner (as in Definition 2.6) is additionally equipped with a counter. The value of counter at the beginning of round k is denoted by γ_k. Now in addition to the properties required for **BasicSwEx**-learnability, we require

(1) $\gamma_0 = \alpha$.

(2) If $r_{k+1} = r_k$, then $\gamma_{k+1} = \gamma_k$.

(3) If $r_{k+1} \neq r_k$, then $\gamma_{k+1} \prec \gamma_k$.

Similarly, one defines the four notions **RestartSw$_*$Ex**, **NewSw$_*$Ex**, **RestartSw$_\alpha$Ex** and **NewSw$_\alpha$Ex** for the restart and newtext switching protocols.

One can consider the generalization of above notions by replacing **Ex** by other convergence criteria such as **BC** [6] or **FEx** [4].

Remark 2.8 The notions, **BasicSwEx**, **RestartSwEx** and **NewSwEx** might change a bit if instead of arbitrary texts some restrictive variants are used.

A *fat text* for a language L, is a text in which every element of L appears infinitely often (and non elements of L never appear). Therefore, arbitrary long initial segments of the text may be missing without losing essential information. For criteria of inference considered in this paper, one may consider learning from "fat information" where all the texts considered in Definition 2.6, are fat texts. In this situation, one can, to a certain degree, compensate the loss of information when switching in the basic switch-protocol. The notions **NewSw$_*$Ex** and **RestartSw$_*$Ex** do not change if one considers fat information, but the notion of **BasicSw$_*$Ex** increases its power and becomes equivalent to **NewSw$_*$Ex** — note that in the standard "non-fat" case by Proposition 3.1 and Theorem 3.2 below, the notion **BasicSw$_*$Ex** is properly contained in **NewSw$_*$Ex**. Similar result applies if one replaces $*$ by an ordinal α in the previous statement.

It can be shown that learning from *recursive texts* does not give any advantage in **TxtEx**-criteria, see, for example, the textbook [13] by Jain, Osherson, Royer and Sharma. All diagonalization results considered in this paper, can be done using recursive texts.

Gold [11] showed that the class of all recursively enumerable sets can be learned from *primitive recursive text*, which are generated by a primitive recursive function. Thus, the generalized criteria considered in this paper coincide with learning from text, if one considers only primitive texts as input in Definition 2.6.

Remark 2.9 Consider the class $\mathcal{L}$ which contains the four subsets of $\{0, 1\}$. This class is **TxtEx**-learnable, but not learnable by a **BasicSwEx**-learner which is required to make *at least* one switch on every possible data-sequence.

To see this, assume that the learner starts with requesting positive examples. As, 0^∞ is a valid text for language $\{0\}$, if the teacher answers 0 on requests for positive examples, eventually the learner must switch and ask for a negative example. Suppose the switch occurs at the n-th round. But then the learner cannot distinguish between the following two situations:

(1) teacher is giving the answers for language $\{0\}$, where $T_+ = 0^\infty$ and $T_- = 1\,2\,3\,\ldots$;

(2) teacher is giving the answers for language $\{0, 1\}$, where $T_+ = 0^n\,1\,0^\infty$ and $T_- = 2\,2\,3\,\ldots$;

the T_- in the above two cases differ at the first position and the T_+ differ at the $(n+1)$-th position. As r_0 was $+$ and r_n was $-$, the learner is not able to distinguish between the two cases.

If the learner starts by requesting negative data, it can be trapped similarly.

As the above remark shows, although **BasicSwEx** is more powerful than **TxtEx**, it still has a severe restriction that information might be lost — it might happen, that a given learner receives, due to switches, a data sequence which satisfies the protocol for several possible languages. This cannot occur for the criteria of **NewSwEx**-learning (for finite number of switches) and **RestartSwEx**-learning (for finite or infinite number of switches), which from this point of view are more natural.

3 Basic Relations between the Concepts

Within this section, we investigate the basic relations between the various criteria of learning by switching type of information.

Proposition 3.1 (a) *For all ordinals* α, $\mathbf{BasicSw_\alpha Ex} \subseteq \mathbf{NewSw_\alpha Ex} \subseteq \mathbf{RestartSw_\alpha Ex}$.
(b) $\mathbf{BasicSw_* Ex} \subseteq \mathbf{NewSw_* Ex} \subseteq \mathbf{RestartSw_* Ex}$.
(c) $\mathbf{BasicSwEx} \subseteq \mathbf{NewSwEx} \subseteq \mathbf{RestartSwEx}$.

Proof. We first show that any teacher using the newtext switch-protocol also satisfies the basic switch-protocol. Thus every learner succeeding with a teacher satisfying the newtext switch-protocol also succeeds with every teacher using the basic switch-protocol. It follows that the inclusion holds for any constraints on the number of switches permitted as the learner does not change.

Consider the interaction between the learner and teacher for any language L. Let r_k denote the request of learner and x_k denote the answer of the teacher in the k-th round, where the answers by the teacher satisfy the newtext switch-protocol. To show that the teacher also satisfies the basic switch-protocol we need to construct texts T_+ (for L) and T_- (for $\overline{L}$) such that $x_k = T_{r_k}(k)$. This can be done by induction. Let $s_+(k)$ and $s_-(k)$ be the number of the $k' < k$ for which $r_{k'}$ is positive or negative, respectively. Now one defines

$$T_+(k) = \begin{cases} x_k & \text{if } r_k = +; \\ s_-(k) & \text{if } r_k = - \text{ and } s_-(k) \in L; \\ \# & \text{if } r_k = - \text{ and } s_-(k) \notin L; \end{cases}$$

$$T_-(k) \;=\; \begin{cases} x_k & \text{if } r_k = -; \\ s_+(k) & \text{if } r_k = + \text{ and } s_+(k) \in \overline{L}; \\ \# & \text{if } r_k = + \text{ and } s_+(k) \notin \overline{L}. \end{cases}$$

Note that all elements of T_+ are either $\#$ or in L since they are either given by the newtext teacher or explicitly required to be in L. Furthermore, if almost all r_k are positive, then the newtext protocol guarantees that all elements of L show up and that T_+ is a text for L. If infinitely many r_k are negative then the function s_- is not bounded and so there is for every $x \in L$ a k such that $x = s_-(k)$ and $r_k = -$. It follows that x goes into the text. Thus T_+ is a text for L and similarly one can verify that T_- is a text for $\overline{L}$.

Also, any teacher using the restart switch-protocol can be used to simulate answers using a newtext switch-protocol – by appropriately repeating the already given positive/negative elements before giving any new elements presented in the restart switch-protocol. The proposition follows. ∎

In the following it is shown that the hierarchy from Proposition 3.1 (c) is strict, that is,

$$\mathbf{TxtEx} \subset \mathbf{BasicSwEx} \subset \mathbf{NewSwEx} \subset \mathbf{RestartSwEx}.$$

Besides this main goal, the influence of restricting the number of switches to be finite or even to respect an ordinal bound, is investigated.

Note that the inclusion $\mathbf{TxtEx} \subseteq \mathbf{BasicSw_0Ex}$ follows directly from the definition. Furthermore, the class $\{L \subseteq \mathbb{N} : \text{card}(\overline{L}) \leq 1\}$, using Proposition 2.4, is not **TxtEx**-learnable; however, as the class contains only cofinite sets, it can be learned via some learner always requesting negative data. Thus the inclusion $\mathbf{TxtEx} \subset \mathbf{BasicSw_0Ex}$ is strict.

Combining finite and cofinite sets is the basic idea to separate newtext switching from basic switching using parts (a) and (c) of Theorem 3.2 below. The class used to show this separation is quite natural:

$$\mathcal{L}_{fin,cofin} = \{L : \text{card}(L) < \infty \text{ or } \text{card}(\overline{L}) < \infty\}.$$

Theorem 3.2 below also characterizes the optimal number of switches needed to learn $\mathcal{L}_{fin,cofin}$ (where possible): one can do it for the criteria $\mathbf{NewSw_*Ex}$ and $\mathbf{RestartSw_*Ex}$ with finitely many switches, but an ordinal bound on the number of switches is impossible.

Theorem 3.2 (a) $\mathcal{L}_{fin,cofin} \in \mathbf{NewSw_*Ex}$.
(b) *For all ordinals α, $\mathcal{L}_{fin,cofin} \notin \mathbf{RestartSw_\alpha Ex}$.*
(c) $\mathcal{L}_{fin,cofin} \notin \mathbf{BasicSwEx}$.

Proof. (a) The machine **M** works in stages. At any point of time it keeps track of elements in L and $\overline{L}$ that it has received.

Construction.

Initially let Pos $= \emptyset$, Neg $= \emptyset$ and go to stage 0.

```
Stage $s$: If card(Pos) $\leq$ card(Neg)

Then request a positive example $x$;
    update Pos $=$ Pos $\cup \{x\} - \{\#\}$;
    conjecture the finite set Pos;

Else request for negative data $x$;
    update Neg $=$ Neg $\cup \{x\} - \{\#\}$;
    conjecture the cofinite set $\mathbb{N} -$ Neg.

Go to stage $s + 1$.

End  stage $s$.
```

It is straight forward to enforce that the learner always represents each conjectured set with the same index. Having this property, it is easy to verify that **M** $\mathbf{NewSw_*Ex}$-learns $\mathcal{L}_{fin,cofin}$.

(b) Suppose by way of contradiction that **M** $\mathbf{RestartSw_\alpha Ex}$-learns the class $\mathcal{L}_{fin,cofin}$. Since every finite sequence of data can be extended to the one of a set in $\mathcal{L}_{fin,cofin}$, **M** has to behave correctly on all data sequences and does not switch without downcounting the ordinal. There is a minimal ordinal β which **M** can reach in some downcounting process. For this β, there is a corresponding round k, a sequence of requests by **M** and a sequence of answers given by a teacher such that **M**'s ordinal counter is β after the k-th round; let Pos be the positive data and Neg be the negative data provided by the teacher until reaching β. As β is minimal, **M** does not make any further downcounting but stabilizes to one type request, say to requesting positive data; the case of requesting only negative data is similar. Let $L = \overline{\text{Neg}}$. If H satisfies $\text{Pos} \subseteq H \subseteq L$ and $\text{card}(L - H) \leq 1$ then **M** is required to learn H without a further switch. So **M** would essentially be a **TxtEx**-learner for $\{H : \text{Pos} \subseteq H \subseteq L \wedge \text{card}(L - H) \leq 1\}$, a contradiction to Proposition 2.4.

(c) Suppose by way of contradiction that **M** **BasicSwEx**-learns $\mathcal{L}_{fin,cofin}$. Due to symmetry-reasons one can assume that the first request of **M** is $+$ and assume that the teacher gives $\#$ as an answer. Now consider the special case that T_- is either $\#^\infty$ or $y\#^\infty$ for some number y. The set to be learned is either $\mathbb{N}$ or $\mathbb{N} - \{y\}$ and the only remaining relevant information is the text T_+. Thus if one could learn $\mathcal{L}_{fin,cofin}$ under the criterion **BasicSwEx**, then one could also **TxtEx**-learn the class $\{L \subseteq \mathbb{N} : \text{card}(\overline{L}) \leq 1\}$, a contradiction to Proposition 2.4. ∎

Item (c) in Theorem 3.2 can be improved to show that even classes which are very easy for **NewSwEx** cannot be **BasicSwEx**-learned.

Corollary 3.3 $\mathbf{NewSw_1Ex} \not\subseteq \mathbf{BasicSwEx}$.

Proof. The proof of Theorem 3.2 (c) shows that the class

$$\{L \subseteq \mathbb{N} : \text{card}(L) \leq 1 \text{ or } \text{card}(\overline{L}) \leq 1\}$$

is not **BasicSwEx**-learnable; the sets with card$(L) \leq 1$ are added for the case that the request r_0 in the proof of Theorem 3.2 (c) is $-$. It remains an easy verification that the considered class is $\mathbf{NewSw_1Ex}$-learnable: A machine first asks for positive examples and outputs an index for the set consisting of the examples seen so far, unless it discovers that there are at least two elements in the language. At which point it switches to requesting negative examples to find the at most one negative example. ▮

The following theorem shows the strength of restarting switch protocol by showing that it has the same learning power as the criterion **InfEx**, where the learner gets the full information on the set L to be learned by reading its characteristic function instead of a text for it, see [11].

Theorem 3.4 **RestartSwEx** = **InfEx**.

Proof. Clearly, **RestartSwEx** $\subseteq$ **InfEx**. In order to show that **InfEx** $\subseteq$ **RestartSwEx**, we show how to construct an informant for the input language using a teacher which follows the restart switch-protocol. Clearly, this suffices to prove the theorem. The learner requests alternatingly, positive and negative information. This gives the learner a text for L as well as for $\overline{L}$, which allows one to construct an informant for the input language L. ▮

The following theorem shows that newtext switching protocols can simulate restart switching protocols, if the number of switches is required to be bounded by an ordinal.

Theorem 3.5 *For all ordinals α,* $\mathbf{RestartSw_\alpha Ex} = \mathbf{NewSw_\alpha Ex}$.

Proof. By Proposition 3.1, it suffices to show the inclusions

$$\mathbf{RestartSw_\alpha Ex} \subseteq \mathbf{NewSw_\alpha Ex}.$$

Note that for $\mathbf{RestartSw_\alpha Ex}$ and $\mathbf{NewSw_\alpha Ex}$ learning, we may assume without loss of generality that the learning machine makes finite number of switches on *all* inputs (i.e., even for inputs for languages outside the class, or for teachers not following the protocol). Furthermore, if the machine makes only finitely many switches, then it is easy to verify that any teacher following the newtext switch-protocol also follows the restart switch-protocol. Theorem follows. ▮

In contrast to Theorem 3.5 the following theorem shows the advantage of restarting switching protocol, compared to newtext switching protocol if the number of switches is not required to be finite.

Theorem 3.6 $\mathbf{RestartSw_* Ex} \not\subseteq \mathbf{NewSwEx}$.

Proof. Let Odd denote the set of odd numbers. Let

$$\begin{aligned}
\mathcal{L}_1 &= \{\text{Odd}\} \cup \{\text{Odd} - \{2x+1\} : x \in \mathbb{N}\},\\
\mathcal{L}_2 &= \{\text{Odd} \cup \{0\}\} \cup \{\text{Odd} \cup \{0\} \cup \{2x+2\} : x \in \mathbb{N}\},\\
\mathcal{L} &= \mathcal{L}_1 \cup \mathcal{L}_2.
\end{aligned}$$

It is easy to see that $\mathcal{L}_1$ can be learned using negative data, and $\mathcal{L}_2$ can be learned using positive data. Thus, a machine can $\mathbf{RestartSw_*Ex}$-identify $\mathcal{L}$ by first finding (by alternatingly requesting positive and negative data) whether 0 belongs to the input language L or not. After this the machine uses just positive data (if $0 \in L$) or just negative data (if $0 \notin L$), to identify L.

To show that $\mathcal{L} \notin \mathbf{NewSwEx}$, we use the fact that any infinite subset of $\mathcal{L}_1$ containing the language Odd, cannot be learned from positive data alone (Proposition 2.4) and that any infinite subset of $\mathcal{L}_2$ containing the language Odd$\cup\{0\}$, cannot be learned from negative data alone (symmetric version of Proposition 2.4).

Suppose by way of contradiction that $\mathcal{L} \in \mathbf{NewSwEx}$ as witnessed by $\mathbf{M}$. Let Even denote the set $\mathbb{N} -$ Odd of even numbers. We then consider the following cases.

Case 1: There exists a way of answering the requests of $\mathbf{M}$ such that positive requests are answered by elements from Odd, negative requests are answered by elements from Even $- \{0\}$ and $\mathbf{M}$ makes infinitely many switches.

In this case, clearly $\mathbf{M}$ cannot distinguish between the cases of input language being Odd and input language being Odd $\cup \{0\}$.

Case 2: Not case 1. Let $x_0, x_1, \ldots, x_k$ be an initial sequence of answers such that

- for $i \leq k$, if $r_i = +$, then $x_i \in$ Odd,
- for $i \leq k$, if $r_i = -$, then $x_i \in$ Even $- \{0\}$,
- if the teacher is consistent with Odd and Odd $\cup \{0\}$, then $\mathbf{M}$ does not make a further switch, that is, the following two conditions hold:
  - if $r_{k+1} = +$ and the teacher takes its future examples $x_{k+1}, x_{k+2}, \ldots$ from the set Odd, then $r_j = r_{k+1}$ for all $j > k$;
  - if $r_{k+1} = -$ and the teacher takes its future examples $x_{k+1}, x_{k+2}, \ldots$ from the set Even $- \{0\}$, then $r_j = r_{k+1}$ for all $j > k$.

Note that there exists such k, $x_0, x_1, \ldots, x_k$, since otherwise one can construct an infinite sequence of answers as needed for case 1, by infinitely often extending a given sequence to force a switch by the learner — leading to infinitely many switches by the learner.

Case 2a: $r_{k+1} = +$.

In this case, $\mathbf{M}$ has to learn the set Odd and every set Odd $- \{2x+1\}$, where $2x+1 \notin \{x_0, x_1, \ldots, x_k\}$, from positive data. This is impossible by Proposition 2.4.

Case 2b: $r_{k+1} = -$.

This is similar to Case 2a. $\mathbf{M}$ needs to learn the set Odd and every set Odd $\cup \{0, 2x\}$, where $2x \notin \{0, x_0, x_1, \ldots, x_k\}$, from negative data. Again this is impossible by symmetric version of Proposition 2.4. ∎

The previous result completes the proof that all inclusions of the hierarchy $\mathbf{TxtEx} \subset \mathbf{BasicSwEx} \subset \mathbf{NewSwEx} \subset \mathbf{RestartSwEx}$ are proper.

4 Counting the Number of Switches

Theorem 4.1 and Corollary 4.2 below show a hierarchy based on the number of switches allowed to the learner.

Theorem 4.1 *For $\alpha \succ \beta$,* $\mathbf{NewSw}_\alpha\mathbf{Ex} \not\subseteq \mathbf{RestartSw}_\beta\mathbf{Ex}$.

Proof. Extend $\prec$ to $\mathrm{Ords} \cup \{-1\}$ by letting $-1 \prec \gamma$, for every $\gamma \in \mathrm{Ords}$. There is a computable function od from $\mathbb{N}$ to $\mathrm{Ords} \cup \{-1\}$ such that

- for every $\gamma \preceq \alpha$ there are infinitely many $x \in \mathbb{N}$ such that $\mathrm{od}(x) = \gamma$;
- there are infinitely many $x \in \mathbb{N}$ such that $\mathrm{od}(x) = -1$;
- the set $\{(x, y) : \mathrm{od}(x) \prec \mathrm{od}(y)\}$ is recursive.

A set $F = \{x_1, x_2, \ldots, x_k\} \subseteq \mathbb{N}$ is α-admissible iff

- $0 < x_1 < x_2 < \ldots < x_k$;
- $\alpha \succ \mathrm{od}(x_1) \succ \mathrm{od}(x_2) \succ \ldots \succ \mathrm{od}(x_k) \succeq -1$.

The empty set is also α-admissible. By definition no infinite set is α-admissible (also note that the second condition postulates a descending chain of ordinals which is always finite). Let the class $\mathcal{L}$ be defined by

$$\begin{aligned} L_F &= \{x : \mathrm{card}(\{0, 1, \ldots, x\} \cap F) \text{ is odd}\}; \\ \mathcal{L}_\alpha &= \{L_F : F \text{ is } \alpha\text{-admissible}\}. \end{aligned}$$

Note that the set $L_\emptyset$ is just $\emptyset$. Intuitively, for $F = \{x_1, x_2, x_3, \ldots, x_k\}$, where $0 < x_1 < x_2 < \ldots < x_k$, one can consider the set of natural numbers to be divided into *blocks*: $B_i = \{x \in \mathbb{N} : x_i \leq x < x_{i+1}\}$, for $i \leq k$, where we take $x_0 = 0$ and $x_{k+1} = \infty$. The odd blocks B_{2i+1} belong to L_F and even blocks B_{2i} belong to $\overline{L_F}$.

Now we show that the class $\mathcal{L}_\alpha$ witnesses the separation.

Claim. $\mathcal{L}_\alpha \in \mathbf{NewSw}_\alpha\mathbf{Ex}$.

Proof of Claim. The machine $\mathbf{M}$ has variables n for the number of switches done so far, E for the finite set of examples seen after the last switch, m_n for the maximal element seen so far and γ_n the value of the ordinal-counter after n switches. The initialization before stage 0 is $E = \emptyset$, $n = 0$, $m_0 = 0$ and $\gamma_0 = \alpha$; $\max_{ordinals} Y$ denotes the maximum element of a non-empty finite set Y of ordinals with respect to their ordering. Intuitively, for $F = \{x_1, x_2, \ldots, x_k\}$, the aim of the algorithm below is to eventually have $m_n \geq x_{k-1}$ (without downcounting the ordinal counter below ordinal 0). It will be shown later that m_n and data of type opposite that of m_n, is enough to identify the language L_F. Go to stage 0.

Construction. Stage s (what is done when the s-th example x is read).

(1) If n is even, request a positive example x;
If n is odd, request a negative example x.

(2) If $x \notin \{\#, 0, 1, \ldots, m_n\}$ and $X = \{y \leq x : 0 \preceq \text{od}(y) \prec \gamma_n\}$ is not empty

Then switch the data type by doing the following:
    Reset $E = \emptyset$;
    Update $n = n + 1$;
    Let $m_n = x$ and $\gamma_n = \max_{ordinals} \{\text{od}(y) : y \in X\}$;
Else let $E = E \cup \{x\} - \{\#\}$.

(3) If $E \nsubseteq \{0, 1, \ldots, m_n\}$,
then let a be the least example outside the set $\{0, 1, \ldots, m_n\}$ which had shown up after the n-th switch
else let $a = m_n$.

(4) If n is even and $a = m_n$ then conjecture E;
If n is even and $a > m_n$ then conjecture $E \cup \{a, a+1, \ldots\}$;
If n is odd and $a = m_n$ then conjecture $\overline{E}$;
If n is odd and $a > m_n$ then conjecture $\overline{E} \cap \{0, 1, \ldots, a\}$.

(5) Go to stage $s + 1$

It is clear that the ordinal is downcounted at every switch of the data presentation. Thus the ordinal bound on the number of mind changes is satisfied.

Assume that F is α-admissible, $k = \text{card}(F)$ and $F = \{x_1, x_2, \ldots, x_k\}$, and the input language is L_F. Let $B_i = \{x \in \mathbb{N} : x_i \leq x < x_{i+1}\}$, for $i \leq k$, where we take $x_0 = 0$ and $x_{k+1} = \infty$.

Below let $\mathbf{n}$ denote the limiting value of n in the above algorithm. At every switch, $\mathbf{M}$ downcounts the ordinal from α through $\gamma_1, \gamma_2, \ldots$ to $\gamma_{\mathbf{n}}$ and thus keeps the ordinal bound. The values $m_0, m_1, \ldots, m_{\mathbf{n}}$ satisfy the condition that $L_F(m_h) \neq L_F(m_{h+1})$, and belong to different block B_i's. Note that the values m_h with odd h are positive and the values m_h with even h are negative examples; $m_0 = 0$ and thus $m_0 \notin L_F$ by definition. Thus, by definition of L_F it follows that $m_1 \geq x_1, m_2 \geq x_2, \ldots, m_{\mathbf{n}} \geq x_{\mathbf{n}}$. By induction, one can easily verify that $\gamma_h \succeq \text{od}(x_h)$, for $h = 1, 2, \ldots, \mathbf{n}$.

After making the $\mathbf{n}$-th switch, $m_{\mathbf{n}}$ has the opposite type of information than the examples seen from then on.

Thus if no information $x > m_{\mathbf{n}}$ arrives after $\mathbf{n}$-th switch, it follows that $x_0, x_1, \ldots, x_k \leq m_{\mathbf{n}}$ and thus every y, such that the type of information of y is opposite to the one of $m_{\mathbf{n}}$, will eventually belong to E. If $\mathbf{n}$ is even then $L_F = E$ (in the limit) and the algorithm is correct. If $\mathbf{n}$ is odd then $L_F = \overline{E}$ (in the limit) and the algorithm is correct again.

If some $x > m_{\mathbf{n}}$ arrives after the last switch, then one knows that $\mathbf{M}$ abstains from switching due to the fact that whenever an example $x > m_{\mathbf{n}}$ arrives then, at step 2, $X = \emptyset$.

Since $\gamma_{\mathbf{n}} \succeq \text{od}(x_{\mathbf{n}}) \succ \text{od}(x_{\mathbf{n}+1})$ and $x_{\mathbf{n}+1} \leq x$, for any $x > m_{\mathbf{n}}$ which arrives after the last switch, we must have that $\text{od}(x_{\mathbf{n}+1}) = -1$, and thus $k = \mathbf{n}+1$. Thus, the least example $a > m_{\mathbf{n}}$ to show up satisfies $a \geq x_k$. Furthermore, every $x \geq a$ satisfies $L_F(x) = L_F(a)$ and it is sufficient to know which of the $x \leq a$ are in L_F and which are not in L_F. This is determined in the limit and thus the sets conjectured by $\mathbf{M}$ are correct.

It is straight forward to ensure that $\mathbf{M}$ always outputs the same index for the same set and thus does not only semantically but also syntactically converge to an index of L_F. ▯

Claim. If a $\mathbf{RestartSw}_\alpha\mathbf{Ex}$-learner $\mathbf{M}$ starts with requesting a negative example first, then $\mathbf{M}$ cannot $\mathbf{RestartSw}_\alpha\mathbf{Ex}$-learn the whole class $\mathcal{L}_\alpha$.

Proof of Claim. Let data of type n be negative data if n is even and positive data if n is odd. So, for this claim, data of type n is what $\mathbf{M}$ requests after n switches. In the following, a set F is constructed such that $\mathbf{M}$ does not $\mathbf{RestartSw}_\alpha\mathbf{Ex}$-learn L_F.

Construction of F. The inductive construction starts with $F = \emptyset$, $n = \text{card}(F)$ and $\mathbf{M}$ requesting examples of type n. There is a finite sequence $\sigma_0\sigma_1 \ldots \sigma_n$ defined inductively such that one of the following cases applies:

Switch: For some σ_n consisting of examples of type n for L_F, $\mathbf{M}$ requests examples of type n after having received $\sigma_0\sigma_1 \ldots \sigma_{n-1}\tau$, for all proper prefix τ of σ_n, but requests example of type $n+1$ after having received $\sigma_0\sigma_1 \ldots \sigma_{n-1}\sigma_n$.

LS: For some σ_n consisting of examples of type n for L_F, $\sigma_0\sigma_1 \ldots \sigma_n$ is a locking-sequence for L_F in the following sense

- for every prefix τ of σ_n, $\mathbf{M}$ requests for example of type n after having received $\sigma_0\sigma_1 \ldots \sigma_{n-1}\tau$, and
- for every extension τ of σ_n, consisting of examples of type n for L_F, $\mathbf{M}$ requests for example of type n after having received $\sigma_0\sigma_1 \ldots \sigma_{n-1}\tau$, and
- for all extensions τ of σ_n, consisting of examples of type n for L_F, $\mathbf{M}$ conjectures L_F as its output after having received $\sigma_0\sigma_1 \ldots \sigma_{n-1}\tau$.

Fail: There is a text T of data of type n for L_F such that for all $\tau \subseteq T$, $\mathbf{M}$, on the sequence $\sigma_0\sigma_1 \ldots \sigma_{n-1}\tau$, requests for example of type n. Furthermore $\mathbf{M}$ on $\sigma_0 \ldots \sigma_{n-1}T$ does not converge to a grammar for L_F.

Note that the above cases are not mutually exclusive. Now the construction of F is continued as follows, based on first case which applies:

Switch: After having seen $\sigma_0\sigma_1 \ldots \sigma_n$, $\mathbf{M}$ downcounts the ordinal to a new value $\gamma' \prec \gamma$. Let x_{n+1} be such that

- $\text{od}(x_{n+1}) = \gamma'$;
- $x_{n+1} > y$ for all $y \in F \cup \text{content}(\sigma_0\sigma_1 \ldots \sigma_n) \cup \{0\}$

and add x_{n+1} to F. Continue the construction with the next inductive step.

LS: Choose a number x_{n+1} such that

- $\mathrm{od}(x_{n+1}) = -1$;
- $x_{n+1} > y$ for all $y \in F \cup \mathrm{content}(\sigma_0\sigma_1 \ldots \sigma_n) \cup \{0\}$

and complete the construction by adding x_{n+1} to F.

Fail: Leave F untouched and complete the construction.

End construction

Verification. Note that in the inductive process, adding a number x_{n+1} to F never makes any previously examples invalid, therefore it is legal to do these modifications during the construction. Furthermore, in the case that it is not possible to satisfy the case "Switch" in the construction at some stage n, one has that after having seen the example-sequence $\sigma_0\sigma_1 \ldots \sigma_{n-1}$ (which is the empty sequence in the case $n = 0$) $\mathbf{M}$ requests only data of type n as long as it sees examples consistent with L_F. Therefore using the locking sequence argument as introduced by Blum and Blum [3], see also [9, 20], either (I) there is a finite sequence σ_n of examples of type n for L_F such that $\sigma_0\sigma_1 \ldots \sigma_n$ is a locking sequence for L_F, that is, case "LS" holds or (II) case "Fail" holds. So it is possible to do the inductive definition in every step.

As the sequence $\mathrm{od}(x_1), \mathrm{od}(x_2), \ldots$ is a falling sequence of ordinals, it must be finite and therefore the construction eventually ends in the cases "LS" or "Fail". In the case "Fail" it is clear that the F constructed gives an L_F not learned by $\mathbf{M}$.

If case LS holds, then let $F' = F - \{x_{n+1}\}$. Note that all $y \geq x_{n+1}$ are examples of type n for $L_{F'}$, and L_F and $L_{F'}$ do not differ on any $z \in \{0, 1, \ldots, x_{n+1} - 1\}$. Thus the information provided to $\mathbf{M}$ is consistent with both L_F and $L_{F'}$. It follows that, given any text T of type n for L_F, $\mathbf{M}$ converges on $\sigma_0\sigma_1 \ldots \sigma_n T$ to an index of $L_{F'}$ and thus does not learn L_F. □

The first claim shows that $\mathcal{L}_\alpha$ is $\mathbf{RestartSw}_\alpha\mathbf{Ex}$-learnable while the second claim shows that such a learner cannot start by requesting a negative example first. Therefore, if $\mathbf{M}$ would be a $\mathbf{RestartSw}_\beta\mathbf{Ex}$-learner for $\mathcal{L}_F$ and $\beta \prec \alpha$, then $\mathbf{M}$ has to start with requesting a positive example. However, then one could consider a new $\mathbf{RestartSw}_\alpha\mathbf{Ex}$-learner $\mathbf{M}'$ which first requests a negative example (without loss of generality assumed to be #), and then switches to positive data, downcounts the ordinal from α to β, and from then on copycats the behaviour of $\mathbf{M}$ with an empty prehistory. It would then follow that $\mathbf{M}$ can $\mathbf{RestartSw}_\beta\mathbf{Ex}$-learn $\mathcal{L}_F$ iff the new learner $\mathbf{M}'$ $\mathbf{RestartSw}_\alpha\mathbf{Ex}$-learns $\mathcal{L}_F$ and starts with requesting a negative example. However this contradicts the second Claim above. Thus no such $\mathbf{M}$ can exists, and the assertion that $\mathcal{L}_F$ witnesses $\mathbf{NewSw}_\alpha\mathbf{Ex} \not\subseteq \mathbf{RestartSw}_\beta\mathbf{Ex}$, for all $\beta \prec \alpha$ is completed. ∎

Corollary 4.2 *Suppose $\alpha \succ \beta$. Then* $\mathbf{BasicSw}_\alpha\mathbf{Ex} \not\subseteq \mathbf{RestartSw}_\beta\mathbf{Ex}$*, in particular:*
(a) $\mathbf{BasicSw}_\alpha\mathbf{Ex} \not\subseteq \mathbf{BasicSw}_\beta\mathbf{Ex}$.
(b) $\mathbf{NewSw}_\alpha\mathbf{Ex} \not\subseteq \mathbf{NewSw}_\beta\mathbf{Ex}$.
(c) $\mathbf{RestartSw}_\alpha\mathbf{Ex} \not\subseteq \mathbf{RestartSw}_\beta\mathbf{Ex}$.

Proof. The main idea is to use the cylindrification $\mathcal{L}_\alpha^{cyl}$ of the class $\mathcal{L}_\alpha$ from Theorem 4.1 in order to show that

$$\mathcal{L}_\alpha^{cyl} \in \mathbf{BasicSw}_\alpha\mathbf{Ex} - \mathbf{RestartSw}_\beta\mathbf{Ex}.$$

Then (a), (b) and (c) follow immediately.

Let $\langle \cdot, \cdot \rangle$ code pairs of natural numbers bijectively into natural numbers: $\langle x, y \rangle = \frac{(x+y)\cdot(x+y+1)}{2} + x$. The cylindrification of a set L is then defined by $L^{cyl} = \{\langle x, y \rangle : x \in L, y \in \mathbb{N}\}$ and $\mathcal{L}_\alpha^{cyl} = \{L^{cyl} : L \in \mathcal{L}_\alpha\}$, where $\mathcal{L}_\alpha$ is as defined in Theorem 4.1.

Note that any text for L^{cyl} ($\overline{L^{cyl}}$) is essentially a fat text for L ($\overline{L}$). Therefore the fact $\mathcal{L}_\alpha \in \mathbf{NewSw}_\alpha\mathbf{Ex}$ implies that $\mathcal{L}_\alpha^{cyl} \in \mathbf{BasicSw}_\alpha\mathbf{Ex}$ by using Remark 2.8. On the other hand, $\mathcal{L}_\alpha^{cyl} \notin \mathbf{RestartSw}_\beta\mathbf{Ex}$ since $\mathcal{L}_\alpha \notin \mathbf{RestartSw}_\beta\mathbf{Ex}$ and by using Remark 2.8 again. ▮

5 Conclusion

The starting point of the present work was the fact that there is a large gap between the data-presentation by a text and by an informant: a text gives only positive data while an informant gives complete information on the set to be learned. So notions of data presentation between these two extreme cases were proposed and the relations between them were investigated. The underlying idea of these notions is that the learner may switch between receiving positive and negative data, but these switches are either finite in number or may cause the loss of information.

For example, the **BasicSwEx**-learner can at every stage only follow one of the texts T_+ and T_- of positive and negative information on the set L to be learned and might therefore miss important information on the other side.

The results of the present work resolve all the relationships between different switching criteria proposed in this paper. In particular it was established that the inclusion

$$\mathbf{TxtEx} \subset \mathbf{BasicSwEx} \subset \mathbf{NewSwEx} \subset \mathbf{RestartSwEx}$$

is everywhere proper. Furthermore, the notion **RestartSwEx** coincides with learning from informant. When we consider restricting the number of switches to meet an ordinal bound, $\mathbf{RestartSw}_\alpha\mathbf{Ex}$ coincides with $\mathbf{NewSw}_\alpha\mathbf{Ex}$. The hierarchy induced by measuring the number of switches with recursive ordinals is proper.

In summary, the notion **NewSwEx** and its variant by bounding the number of switches turned out to be the most natural definition in the gap between **TxtEx**-learning and learning from informant. The notion of **BasicSwEx**-learning is between **TxtEx**-learning and learning from informant, but has some strange side-effects: requiring some minimum number of switches may be more harmful than requiring no switches, as pointed out in Remark 2.9. On the other hand, **RestartSwEx** coincides with other notions, as mentioned above.

Note that these criteria differ from learning from negative open text as considered by Baliga, Case and Jain [2], this is notion (b) from the introduction. Learning from open negative text is weaker than learning from informant and thus different from **RestartSwEx**.

On the other hand, the class $\mathcal{L}_{fin,cofin}$ and the class $\mathcal{L}$ from Theorem 3.6 are both learnable from negative open text and so separate this notion from the other switching criteria mentioned in this paper.

There is an application of learning by switching type of information to the field of learning algebraic substructures of vector spaces. Harizanov and Stephan [12] investigated when it is possible to learn the class $\mathcal{L}$ of all recursively enumerable subspaces of the space V_∞/V. Here V_∞ is the standard recursive vector space over the rationals with countably infinite dimension and V is a given recursively enumerable subspace of V_∞. The space V_∞/V is called k-thin iff there is a subspace $W \in \mathcal{L}$ such that V/W is k-dimensional and, for every $U \in \mathcal{L}$, U is an infinite dimensional subspace of V_∞/V iff $W \subseteq U$. While $\mathcal{L}$ is **TxtBC**-learnable iff V_∞/V is finite dimensional, $\mathcal{L}$ is **NewSwBC**-learnable iff either $\mathcal{L}$ is already **TxtBC**-learnable or V_∞/V is 0-thin or 1-thin. **InfBC**-learning is much more powerful; it covers the case of all k-thin spaces, but there is no effective algebraic characterization of the spaces where $\mathcal{L}$ is learnable. So **NewSwBC**-learning turned out to be the only notion where learnability of the class of recursively enumerable subspaces has an interesting and non-trivial algebraic characterization.

References

[1] Dana Angluin. Learning regular sets from queries and counter-examples. *Information and Computation*, 75:87–106, 1987.

[2] Ganesh Baliga, John Case and Sanjay Jain. Language learning with some negative information. *Journal of Computer and System Sciences*, 51(5):273–285, 1995.

[3] Lenore Blum and Manuel Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.

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

[5] John Case, Sanjay Jain and Mandayam Suraj. Not-so-nearly-minimal-size program inference. In K. Jantke and S. Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 77–96. Springer-Verlag, 1995.

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

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

[8] Rūsiņš Freivalds and Carl Smith. On the role of procrastination in machine learning. *Information and Computation*, 107:237–271, 1993.

[9] Mark Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.

[10] William Gasarch and Mark Pleszkoch. Learning via queries to an oracle. In R. Rivest, D. Haussler, and M. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory*, pages 214–229. Morgan Kaufmann, 1989.

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

[12] Valentina Harizanov and Frank Stephan. *On the Learnability of Vector Spaces.* Forschungsberichte Mathematische Logik 55/2002, Mathematical Institute, University of Heidelberg, 2002.

[13] Sanjay Jain, Daniel Osherson, James Royer and Arun Sharma. *Systems that Learn: An Introduction to Learning Theory.* MIT Press, Cambridge, Mass., second edition, 1999.

[14] Sanjay Jain and Arun Sharma. On the non-existence of maximal inference degrees for language identification. *Information Processing Letters*, 47:81–88, 1993.

[15] Sanjay Jain, Wolfram Menzel and Frank Stephan. Classes with easily learnable subclasses. In *Algorithmic Learning Theory: Thirteenth International Conference* (*ALT 2002*), 2002. To appear.

[16] Stephen Kleene. Notations for ordinal numbers. *The Journal of Symbolic Logic*, 3:150–155, 1938.

[17] Micheal Machtey and Paul Young. *An Introduction to the General Theory of Algorithms.* North Holland, New York, 1978.

[18] Wolfgang Merkle and Frank Stephan. Refuting learning revisited. In *Algorithmic Learning Theory: Twelfth International Conference* (*ALT 2001*), volume 2225 of *Lecture Notes in Artificial Intelligence*, pages 299–314. Springer-Verlag, 2001.

[19] Tatsuya Motoki. Inductive inference from all positive and some negative data. *Information Processing Letters*, 39(4):177–182, 1991.

[20] Daniel Osherson, Michael Stob and Scott Weinstein. *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists.* MIT Press, 1986.

[21] Hartley Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, 1967. Reprinted, MIT Press 1987.

[22] Gerald E. Sacks. *Higher Recursion Theory.* Springer-Verlag, 1990.

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