Results on Memory-Limited U-Shaped Learning

Lorenzo Carlucci[⋆1] John Case[⋆⋆2], Sanjay Jain[⋆⋆⋆3], Frank Stephan[†4]

[1] Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716-2586, USA and Dipartimento di Matematica, Università di Siena, Pian dei Mantellini 44, Siena, Italy, EU.
carlucci5@unisi.it

[2] Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716-2586, USA.
case@cis.udel.edu

[3] School of Computing, 3 Science Drive 2, National University of Singapore, Singapore 117543, Republic of Singapore.
sanjay@comp.nus.edu.sg

[4] School of Computing and Department of Mathematics, National University of Singapore, 3 Science Drive 2, Singapore 117543, Republic of Singapore.
fstephan@comp.nus.edu.sg

Abstract. U-shaped learning is a learning behaviour in which the learner first *learns* a given target behaviour, then *unlearns* it and finally *relearns* it. Such a behaviour, observed by psychologists, for example, in the learning of past-tenses of English verbs, has been widely discussed among psychologists and cognitive scientists as a fundamental example of the non-monotonicity of learning. Previous theory literature has studied whether or not U-shaped learning, in the context of Gold's formal model of learning languages from positive data, is *necessary* for learning some tasks.

It is clear that human learning involves memory limitations. In the present paper we consider, then, the question of the necessity of U-shaped learning for some learning models featuring *memory limitations*. Our results show that the question of the necessity of U-shaped learning in this memory-limited setting depends on delicate tradeoffs between the learner's ability to remember its own previous conjecture, to store some values in its long-term memory, to make queries about whether or not items occur in previously seen data *and* on the learner's choice of hypothesis space.

1 Introduction and Motivation

In Section 1.1 we explain U-shaped learning and in Section 1.2 memory-limited learning. In Section 1.3 we summarize our *main* results of the present paper with pointers to later sections where they are treated in more detail.

1.1 U-Shaped Learning

U-shaped learning occurs when the learner first learns a correct behaviour, then abandons that correct behaviour and finally returns to it once again. This pattern of learning has been observed by cognitive and developmental psychologists in a variety of child development phenomena, such as language learning [8, 29, 42], understanding of temperature [42, 43], understanding of weight conservation [7, 42], object permanence [7, 42] and face recognition [9]. The case of language

⋆ Supported in part by NSF grant number NSF CCR-0208616.

⋆⋆ Supported in part by NSF grant number NSF CCR-0208616.

⋆⋆⋆ Supported in part by NUS grant number R252–000–127–112 and R252–000–212–112.

† Supported in part by NUS grant number R252–000–212–112.

acquisition is paradigmatic. In the case of the past tense of English verbs, it has been observed that children learn correct syntactic forms (call/called, go/went), then undergo a period of overregularization in which they attach regular verb endings such as ‘ed’ to the present tense forms even in the case of irregular verbs (break/breaked, speak/speaked) and finally reach a final phase in which they correctly handle both regular and irregular verbs. This example of U-shaped learning behaviour has figured so prominently in the so-called “Past Tense Debate” in cognitive science that competing models of human learning are often judged on their capacity for modeling the U-shaped learning phenomenon [29, 36, 44]. Recent interest in U-shaped learning is also witnessed by the fact that the *Journal of Cognition and Development* dedicated its first issue in the year 2004 to this phenomenon.

While the prior cognitive science literature on U-shaped learning was typically concerned with modeling *how* humans achieve U-shaped behaviour, [2, 10] are motivated by the question of *why* humans exhibit this seemingly inefficient behaviour. Is it a mere harmless evolutionary inefficiency or is it *necessary* for full human learning power? A technically answerable version of this question is: are there some formal learning tasks for which U-shaped behaviour is logically necessary? The answer to this latter question requires that we first describe some formal criteria of successful learning.

A learning machine $\mathbf{M}$ reads an infinite sequence consisting of the elements of any language L in arbitrary order with possibly some pause symbols # in between elements. During this process the machine outputs a corresponding sequence $e_0\, e_1\, \ldots$ of hypotheses (grammars) which may generate the language L to be learned. Sometimes, especially when numerically coded, we also call these hypotheses *indices*. A fundamental criterion of successful learning of a language is called *explanatory learning* (**Ex**-*learning*) and was introduced by Gold [20]. Explanatory learning requires that the learner’s output conjectures stabilize in the limit to a *single* conjecture (grammar/program, description/explanation) that generates the input language. *Behaviourally correct learning* [14, 33] requires, for successful learning, only convergence in the limit to possibly infinitely many syntactically distinct but correct conjectures. Another interesting class of criteria features *vacillatory learning* [12, 21]. This paradigm involves learning criteria which allow the learner to vacillate in the limit between *at most* finitely many syntactically distinct but correct conjectures. For each criterion that we consider above (and below), a *non U-shaped learner* is naturally modeled as a learner that never *semantically* returns to a previously abandoned correct conjecture on languages it learns according to that criterion.

Baliga and his co-workers [2] showed that every **Ex**-learnable class of languages is **Ex**-learnable by a non U-shaped learner, that is, for **Ex**-learnability, U-shaped learning is *not* necessary. Furthermore, based on a proof of Fulk, Jain and Osherson [19], Baliga and his co-workers [2] noted that, by contrast, for behaviourally correct learning, U-shaped learning *is* necessary for full learning power. In [10] it is shown that, for non-trivial vacillatory learning, U-shaped learning is again necessary (for full learning power).

1.2 Memory-Limited Learning

It is clear that human learning involves memory limitations. In the present paper we consider the necessity of U-shaped learning in formal *memory-limited* versions of language learning. In the prior literature at least the following three types of memory-limited learning have been studied.

A most basic concept of memory-limited learning is *iterative learning* [27, 46, 47], according to which the learner reacts to its current data item, can remember its own last conjecture but cannot store *any* of the strictly previously seen data items. Iterative learning admits of learning non-trivial classes. For example, the class of finite sets is iteratively learnable as is a class of self-describing sets, for example, the class of languages with the least element coding a grammar for the language. Furthermore, for each $m \geq 1$, the class of unions of m of Angluin's [1] pattern languages is iteratively learnable [13]. The notion of *n-feedback learning* denotes iterative learning where, in addition, the learner can make n simultaneous queries asking whether some datum has been seen in the past [13, 27]. Finally, a learner is called an *n-bounded example memory* learner [13, 19, 27, 34] if, besides reacting to its currently seen data item and remembering its own last conjecture, it is allowed to store in "long term memory" at most n strictly previously seen data items.

For the present paper, our first intention was to study the impact of forbidding U-shaped learning in each of the above three models of memory-limited learning. So far we have had success for these problems only for some more restricted variants of the three models. Hence, we now describe these variants.

Our variants of iterative learning are motivated by two aspects of Gold's model. The first aspect is the absolute freedom allowed regarding the *semantic* relations between successive conjectures, and between the conjectures and the input. Many forms of semantic constraints on the learner's sequence of hypotheses have been studied in the previous literature (for example, conservativity [1], consistency [1, 4], monotonicity [23, 48], set-drivenness [12, 18, 46]) and it is reasonable to explore their interplay with U-shaped learning in the memory-bounded setting of iterative learning. Secondly, it is well-known that the choice of the hypothesis space from which the learner can pick its conjectures has an impact on the learning power [26, 27]. We accordingly also consider herein U-shaped iterative learning with restrictions on the hypothesis space.

For the case of feedback learning, we introduce and consider a model called *n-memoryless feedback learning* which restricts n-feedback learning so that the learner does *not* remember its last conjecture. These criteria form a hierarchy of more and more powerful learning criteria increasing in n and, for $n > 0$, are incomparable to iterative learning, see Theorem 32 and Remark 25 each in Section 6. The criterion of 0-memoryless feedback learning is properly contained in the criterion of iterative learning, see Remark 38 in Section 7 below.

Finally, in Section 7, we introduce a more limited variant of bounded example memory, *c-bounded memory states learning* for which the learner does not remember its previous conjecture *but* can store any one out of c different *values* in its long term memory [16, 17, 24]. For example, when $c = 2^k$, the memory is equivalent to k bits of memory. By Theorem 37, these criteria form a hierarchy of more and more powerful learning criteria increasing in c. Furthermore, the comparisons between bounded memory states learning, iterative learning and memoryless feedback learning are presented in Remark 38.

Our results herein on memory-limited models are presented for **Ex**-*learning*. This is, in part, justified by the following considerations. In Section 3, Propositions 7 and 8 essentially imply that, for iterative learning, the **Ex** case is the only interesting case. In Section 7, Theorem 34 implies that c-bounded memory states behaviourally correct learning can be replaced by $(c+1)!$-bounded memory states **Ex**-learning.

1.3 Brief Summary of Main Results

In Section 3 we formally define *iterative learning* and prove some background facts about it. Furthermore, we state the basic connections between iterative non U-shaped learning and iterative U-shaped learning in the context of behaviourally correct, vacillatory and explantory learning.

In Section 4 we study the interplay of hypotheses spaces and non U-shaped learning. An *indexed family of recursive languages* $\mathcal{L}$ is a class of recursive languages $L_0, L_1, L_2, \ldots$ such that the predicate $x \in L_i$ is uniformly recursive in both i and x. In this context, i is called an *index* of L_i; this i codes how to algorithmically decide L_i. Angluin [1] noticed the importance of indexed families for learning theory, gave a characterization when such a class is learnable from positive data and stated that many classes considered in learning theory are indeed indexed families. *Class-preserving language learning* by a learner $\mathbf{M}$ [26] of an indexed family $\mathcal{L}$ is $\mathbf{Ex}$-learning, where, instead of using an acceptable programming system for making its conjectures, the learner uses *some* recursive indexing of $\mathcal{L}$. In particular, the main result of Section 4, Theorem 12, shows that U-shaped learning is necessary for the full learning power of class-preserving iterative learning [27].

In Section 5 we study, in the context of iterative learning, the relation of the non U-shapedness constraint to other well studied constraints on the *semantic* behaviour of the learner's conjectures. We consider *class-consistent learning* [1, 4], according to which the learner's conjectures, on the languages it learns, must generate all the data on which they are based. *Monotonic learning* by a machine $\mathbf{M}$ [48] requires that, on any input language L that $\mathbf{M}$ $\mathbf{Ex}$-learns, a new hypothesis cannot reject an element $x \in L$ that a previous hypothesis already included. Theorem 19 shows that class-consistent iterative learners can be turned into iterative non U-shaped *and* monotonic learners.

In Section 6, we formally define *n-memoryless feedback learning* (discussed near the end of Section 1.2 above) and consider the impact of forbidding U-shaped learning in this setting. The main result of Section 6, Theorem 30, shows that U-shaped learning *is necessary* for the full learning power of n-memoryless feedback learners.

In Section 7 we formally introduce *c-bounded memory states learning* (also discussed near the end of Section 1.2 above). The main result of this section, Theorem 35, shows that U-shaped behaviour does *not* enhance the learning power of 2-bounded memory states learners. Here the memory in the $c = 2$ case is 1 bit of memory; it is open as to how Theorem 35 goes for c-bounded memory states learners, where $c > 2$.

In Section 8 we summarize and briefly discuss our main results, and collect open problems.

2 Notation and Preliminaries

2.1 Recursion Theory Background

Any unexplained recursion theoretic notation is from [37]. For general background on Recursion Theory we refer the reader to the standard text books [31, 32, 37, 41]. The symbol $\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. The 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 $\mathrm{card}(S)$. $\mathrm{card}(S) \leq *$ denotes that S is finite. The maximum and minimum of a set are denoted by $\max(\cdot), \min(\cdot)$, respectively, where $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$.

We let $\langle \cdot, \cdot \rangle$ stand for Cantor's computable, bijective mapping $\langle x, y \rangle = \frac{1}{2}(x+y)(x+y+1)+x$ from $\mathbb{N} \times \mathbb{N}$ onto $\mathbb{N}$ [37]. Note that $\langle \cdot, \cdot \rangle$ is monotonically increasing in both of its arguments. We define $\pi_1(\langle x, y \rangle) = x$ and $\pi_2(\langle x, y \rangle) = y$.

By φ we denote a fixed *acceptable numbering* (programming system) [37] for the partial-recursive functions mapping $\mathbb{N}$ to $\mathbb{N}$. By φ_i we denote the partial-recursive function computed by the program with number i in the φ-system. By Φ we denote an arbitrary fixed Blum complexity measure [6] for the φ-system. A partial recursive function $\Phi(\cdot,\cdot)$ is said to be a Blum complexity measure for φ, iff the following two conditions are satisfied:

- for all i and x, $\Phi(i,x)\downarrow$ iff $\varphi_i(x)\downarrow$.
- the predicate $P(i,x,t) \equiv \Phi(i,x) \leq t$ is decidable.

By convention we use Φ_i to denote the partial recursive function $x \to \Phi(i,x)$. Intuitively, $\Phi_i(x)$ may be thought as the number of steps it takes to compute $\varphi_i(x)$.

By W_i we denote the domain of φ_i. That is, W_i is then the recursively enumerable (r.e.) subset of $\mathbb{N}$ accepted by the φ-program i. Note that all acceptable numberings are recursively isomorphic and that one therefore could also define W_i to be the set generated by the i-th grammar. The symbol L ranges over the class of r.e. sets. By $\overline{L}$, we denote the complement of L, that is $\mathbb{N} - L$. The symbol $\mathcal{L}$ ranges over classes containing some but not necessarily all r.e. sets. By $W_{i,s}$ we denote the set $\{x \leq s : \Phi_i(x) \leq s\}$. Similarly, $\varphi_{i,s}(x)$ denotes $\varphi_i(x)$ if $x \leq s$ *and* $\Phi_i(x) \leq s$; otherwise $\varphi_{i,s}(x)$ is undefined.

2.2 Explanatory and Non-U-Shaped Learning

We now present concepts from language learning theory [20, 21]. The next definition introduces the concept of a *sequence* of data.

Definition 1. (a) A *sequence* σ is a mapping from an initial segment of $\mathbb{N}$ into $(\mathbb{N} \cup \{\#\})$. The empty sequence is denoted by λ.

(b) The *content* of a sequence σ, denoted content(σ), is the set of natural numbers in the range of σ.

(c) The *length* of σ, denoted by $|\sigma|$, is the number of elements in σ. So, $|\lambda| = 0$.

(d) For $n \leq |\sigma|$, the initial sequence of σ of length n is denoted by $\sigma[n]$. So, $\sigma[0]$ is λ.

Intuitively, the pause-symbol # represents a pause in the presentation of data. We let σ, τ and γ range over finite sequences. We denote the sequence formed by the concatenation of τ at the end of σ by $\sigma\tau$. Sometimes we abuse the notation and use σx to denote the concatenation of sequence σ and the sequence of length 1 which contains the element x. $(\mathbb{N} \cup \{\#\})^*$ denotes the set of all finite sequences.

Quite frequently used in this paper is the existence of a one-one recursive function pad(e, X) with $W_{\text{pad}(e,X)} = W_e$, where — according to the context — X might be a number, a finite set or a finite sequence. In particular, pad is chosen such that e, X can be computed from pad(e, X) by a recursive function.

Definition 2. [20] (a) A *text* T for a language L is a mapping from $\mathbb{N}$ into $(\mathbb{N} \cup \{\#\})$ such that L is the set of natural numbers in the range of T. $T(i)$ represents the $(i+1)$-th element in the text.

(b) The *content* of a text T, denoted by content(T), is the set of natural numbers in the range of T; that is, the language which T is a text for.

(c) $T[n]$ denotes the finite initial sequence of T with length n.

We now define the basic paradigm of learning in the limit, explanatory learning.

Definition 3. [20] A learner $\mathbf{M} : (\mathbb{N}\cup\{\#\})^* \to (\mathbb{N}\cup\{?\})$ is a (possibly partial) recursive function which assigns hypotheses to finite strings of data. $\mathbf{M}$ **Ex**-learns a class $\mathcal{L}$ (equivalently $\mathbf{M}$ is an **Ex**-learner for $\mathcal{L}$) iff for every $L \in \mathcal{L}$ and every text T for L, $\mathbf{M}$ is defined on all initial segments of T, and there is an index n such that $\mathbf{M}(T[n]) \neq ?$, $W_{\mathbf{M}(T[n])} = L$ and $\mathbf{M}(T[m]) \in \{\mathbf{M}(T[n]), ?\}$ for all $m \geq n$. **Ex** denotes the collection of all classes of languages that can be **Ex**-learned.

For **Ex**-learnability one may assume without loss of generality that the learner is total. However, for some of the criteria below, such as class consistency and iterative learning, this cannot be assumed without loss of generality. The requirement for $\mathbf{M}$ to be defined on each initial segment of each text for a language in $\mathcal{L}$ is also assumed for learners with other criteria considered below.

Now we define non U-shaped learning. A non U-shaped learner never makes the sequence correct–incorrect–correct while learning a language that it actually learns. Thus, since such a learner has eventually to be correct, one can make the definition a bit simpler than the idea behind the notion suggests.

Definition 4. [2]

(a) We say that $\mathbf{M}$ is non U-shaped on text T, if for all n and $m > n$, if $W_{\mathbf{M}(T[n])} = \text{content}(T)$, then ($\mathbf{M}(T[m]) = ?$ or $W_{\mathbf{M}(T[m])} = \text{content}(T)$).

(b) We say that $\mathbf{M}$ is non U-shaped on L if $\mathbf{M}$ is non U-shaped on each text for L.

(c) We say that $\mathbf{M}$ is non U-shaped on $\mathcal{L}$ if $\mathbf{M}$ is non U-shaped on each $L \in \mathcal{L}$.

Definition 5. Let $\mathbf{I}$ be a learning criterion. Then $\mathbf{NUShI}$ denotes the collection of all classes $\mathcal{L}$ such that there exists a machine $\mathbf{M}$ that learns $\mathcal{L}$ according to $\mathbf{I}$ and is non U-shaped on $\mathcal{L}$.

3 Iterative Learning

The **Ex**-model makes the assumption that the learner has access to the full history of previous data. On the other hand it is reasonable to think that humans have more or less severe memory limitations. This observation motivates, among other criteria discussed in the present paper, the concept of *iterative learning*. An iterative learner features a severe memory limitation: it can remember its own previous conjecture but not its *past* data items. Moreover, each conjecture of an iterative learner is determined as an algorithmic function of the previous conjecture *and* of the current input data item.

The formal definition of an iterative learner is the following.

Definition 6. [46, 47] An iterative learner is a (possibly partial) function $\mathbf{M} : (\mathbb{N} \cup \{?\}) \times (\mathbb{N} \cup \{\#\}) \to (\mathbb{N} \cup \{?\})$ together with an initial hypothesis $e_0 \in \mathbb{N} \cup \{?\}$. $\mathbf{M}$ **It**-learns a class $\mathcal{L}$ iff, for every $L \in \mathcal{L}$ and every text T for L, the sequence $e_0, e_1, \ldots$ defined inductively by the rule $e_{n+1} = \mathbf{M}(e_n, T(n))$ satisfies: there exists an m such that e_m is an index for L and for all $n \geq m$, $e_n \in \{e_m, ?\}$. **It** denotes the collection of all iteratively learnable classes.

For iterative learners (without other constraints), one may assume without loss of generality that they never output ?.

It is well-known that $\mathbf{It} \subset \mathbf{Ex}$ [47]. One might therefore ask whether iterative learning is also restrictive in the case of behaviourally correct convergence, which allows the learner to converge in the limit to possibly infinitely many syntactically distinct correct conjectures. That this is not the case can be easily shown using padding, that is, a one-to-one recursive function pad with $W_{\text{pad}(e,\sigma)} = W_e$ for all strings σ. Given any behaviourally correct learner $\mathbf{M}$ for a class $\mathcal{L}$, one can define a new learner $\mathbf{N}$ on input σ implicitly as $\text{pad}(\mathbf{M}(\sigma), \sigma)$. This new learner can explicitly be defined as an iterative learner by starting with $\text{pad}(\mathbf{M}(\emptyset), \lambda)$ and updating via

$$\mathbf{N}(\text{pad}(e, \sigma), x) = \text{pad}(\mathbf{M}(\sigma x), \sigma x)$$

where it has the full access to the previous data since it codes this information into the index. Thus it can reconstruct the hypothesis $\mathbf{M}(\sigma x)$ from the old hypothesis $\text{pad}(e, \sigma)$ and the new datum x. This simple argument proves the following Proposition.

Proposition 7. *Every behaviourally correct learnable class has an iterative behaviourally correct learner.*

By Proposition 7 it does not make sense to consider behaviourally correct iterative learning. So one might look at restrictions of behaviourally correct learning like the notion of vacillatory learning [12]. The next result shows that relaxing the convergence requirement of iterative learning to vacillatory convergence does not increase learnability at all.

Proposition 8. *If some iterative learner* $\mathbf{M}$ *eventually vacillates on every text of every language in* $\mathcal{L}$ *between finitely many correct hypotheses, then* $\mathcal{L} \in \mathbf{It}$.

Proof. Without loss of generality assume that $\mathbf{M}$ does not output ?. Given $\mathbf{M}$ as above, one defines $\mathbf{N}$ as follows. $\mathbf{N}$ will output grammars of form: $\text{pad}(p, S)$, where S is a finite set (not containing p), and $\text{pad}(p, S)$ is a padding function such that $\text{pad}(p, S)$ is a grammar for W_p, and p, S can be extracted from $\text{pad}(p, S)$. Let the initial hypothesis of $\mathbf{N}$ be $\text{pad}(\mathbf{M}(\lambda), \emptyset)$ and the update rule be

$$\mathbf{N}(\text{pad}(p, S), x) = \begin{cases} \text{pad}(p, S), & \text{if } \mathbf{M}(p, x) \in \{p\} \cup S; \\ \text{pad}(\mathbf{M}(p, x), S \cup \{p\}), & \text{otherwise.} \end{cases}$$

We claim that $\mathbf{N}$ is an iterative learner for $\mathcal{L}$.

In the following, by "last brand new hypothesis output by $\mathbf{M}$ on σ", we mean: let $\tau \subseteq \sigma$ be such that: $\mathbf{M}$, after seeing τ, outputs q; $\mathbf{M}$ does not output q after seeing any proper initial segment of τ; on any $\gamma \subseteq \sigma$, the output of $\mathbf{M}$ after seeing γ is same as output of $\mathbf{M}$ after seeing γ', for some $\gamma' \subseteq \tau$.

For any sequence σ, we will define a derived sequence τ_σ below satisfying the following four conditions.

(I) $\text{content}(\tau_\sigma) = \text{content}(\sigma)$.
(II) If $\sigma \subseteq \sigma'$ then $\tau_\sigma \subseteq \tau_{\sigma'}$.
(III) If the last output of $\mathbf{N}$ after seeing σ is $\text{pad}(p, S)$, then the last output of $\mathbf{M}$ after seeing τ_σ is p, and the set of programs output by $\mathbf{M}$ on initial segments of τ_σ is $\{p\} \cup S$.
(IV) The last brand new hypothesis output by $\mathbf{M}$ on τ_σ is p.

The above properties will be inductively seen to be true, based on length of σ.

Base Case: $\tau_\lambda = \lambda$. Clearly, properties (I)—(IV) hold for the base case.

Inductive case: Suppose we have defined τ_σ. Define $\tau_{\sigma x}$ as follows. Suppose $\mathbf{N}$ after seeing σ outputs pad(p, S). Thus, $\mathbf{M}$ on initial segments of τ_σ, would have output programs from $S \cup \{p\}$, with the output after seeing τ_σ being p (by induction).

If $\mathbf{M}(p, x)$ is a program not in $S \cup \{p\}$, then let $\tau_{\sigma x} = \tau_\sigma x$. Else, let γ be initial segment of τ_σ such that $\mathbf{M}$, after seeing γ, had output $\mathbf{M}(p, x)$. Let γ' be such that $\tau_\sigma = \gamma\gamma'$. Then, $\tau_{\sigma x} = \tau_\sigma x \gamma'$. It is easy to verify that properties (I)–(IV) hold in both cases.

Let T be a text for $L \in \mathcal{L}$. Let $\tau_T = \bigcup_{n \in \mathbb{N}} \tau_{T[n]}$. It is easy to verify using property (I) that content(τ_T) = content(T). Also, since $\mathbf{M}$ eventually vacillates between finitely many correct hypotheses on τ_T, by property (IV), $\mathbf{N}(T)$ converges to pad(p, S), where p is the last brand new hypothesis output by $\mathbf{M}$ on τ_T, and S is the set of hypothesis output by $\mathbf{M}$ on τ_T, except for the grammar p. Furthermore, by property (III), p is output by $\mathbf{M}$ on τ_T infinitely often. It follows that $\mathbf{M}$ learns L from T. ∎

Thanks to Propositions 7 and 8 we will, from now on, consider *explanatory* iterative learners only. All our notions regarding iterative learning will be modifications of the basic **Ex**-learning paradigm. An important question in this context was whether iterative explanatory learning needs U-shapedness for full learning power. Quite recently, Samuel Moelius [30] solved this important open problem by showing that iterative learning coincides with its non U-shaped variant.

Theorem 9. [30] *Every iteratively* **Ex***-learnable class has a non U-shaped iterative* **Ex***-learner.*

Together with Proposition 8, one has the following corollary.

Corollary 10. *If some iterative learner* **M** *eventually vacillates on every text of every language in $\mathcal{L}$ between finitely many correct hypotheses then $\mathcal{L}$ has a non U-shaped iterative and explanatory learner.*

Our goal is to look at the corresponding question for several related notions, but before doing this, we want to conclude this section by briefly recalling some basic relations of iterative learning with two criteria of learning that feature, like non U-shaped learning, a semantic constraint on the learner's sequence of hypotheses.

The first such notion is *set-driven learning* [46], where the hypotheses of a learner on inputs σ, τ are the same whenever content(σ) = content(τ). We denote by **SD** the collection of all classes learnable by a set-driven learner. It is shown in [24, Theorem 7.7] that $\mathbf{It} \subseteq \mathbf{SD}$. The inclusion is proper since the class of all finite supersets of $\{0\}$ plus the set $\{1, 2, 3, \ldots\}$ has a set-driven but no iterative learner.

A criterion that implies non U-shapedness is *conservative learning* [1]. A learner is conservative iff whenever it makes a mind change from a hypothesis i to j then it has already seen some datum $x \notin W_i$. **Consv** denotes the collection of all classes having a conservative learner.

It is shown in [24] that $\mathbf{SD} \subseteq \mathbf{Consv}$, thus, $\mathbf{It} \subset \mathbf{Consv}$. By definition, every hypothesis abandoned by a conservative learner is incorrect and thus $\mathbf{Consv} \subseteq \mathbf{NUShEx}$ follows. It is well known that the latter inclusion is proper. The easiest way to establish it is to use Angluin's proper inclusion $\mathbf{Consv} \subset \mathbf{Ex}$ [1] and the equality from $\mathbf{Ex} = \mathbf{NUShEx}$ [2].

4 Iterative Learning and Hypothesis Spaces

Normally, in Gold-style language learning, a learner outputs as hypotheses just indices from a fixed acceptable enumeration of all r.e. languages, since all types of output (programs, grammars and so on) can be translated into these indices. But there have also been investigations [1, 26, 27] where the hypothesis space is fixed in the sense that the learner has to choose its hypotheses either from this fixed space (exact learning) or from a space containing exactly the same languages (class-preserving learning).

Such a restriction can be severe. For example, the class of all finite sets is iteratively learnable and so is the class $\mathcal{L}$ of all finite sets of even cardinality. But if one requires the hypotheses to be from some one-one enumeration of $\mathcal{L}$, then one forces the learner to output indices which do not uniquely encode information on which data have been seen so far. This imposes some forgetting which can be used to show that the class $\mathcal{L}$ is not exactly iteratively learnable when the underlying hypothesis space is one-one.

In this section we investigate ways in which the hypothesis space interferes with non U-shaped iterative learnability.

To explain our first result we need to recall some notions of computations relative to oracles [37]. Let A be a set. A partial function f is called *computable relative to* A if there is an algorithm for f that is allowed to use answers to questions of the form '$x \in A$?'. A is then called an *oracle*. A total function which is computable relative to A is also referred to as *an A-recursive function*; a set B is called *A-recursive* if the characteristic function of B is A-recursive. We say that B is Turing reducible to A, written $B \leq_T A$, in this case.

There exists an acceptable enumeration of all partial functions computable relative to A. Let $\varphi_0^A, \varphi_1^A, \ldots$ be such an enumeration. W_e^A denotes the domain of φ_e^A. A Blum complexity measure $\Phi^A(\cdot, \cdot)$ can be defined for φ^A as a partial function computable in A, see [28, 39]. The function $x \mapsto \Phi^A(i, x)$ is partial computable relative to A and can be intuitively seen as the number of steps needed to compute $\varphi_i^A(x)$. The predicate $\Phi^A(i, x) \leq t$ will be no longer recursive but total computable in A instead. We use $\Phi_i^A(x)$ to denote $\Phi^A(i, x)$. Let K be the diagonal halting problem $\{x \in \mathbb{N} : x \in W_x\}$. Recall that K is an r.e. set which is not recursive. Given a set A, one can consider the diagonal halting problem for the partial functions that are computable with oracle A. Then A' denotes this diagonal halting problem relativized to A, that is, the set $\{x \in \mathbb{N} : x \in W_x^A\}$. A' is called *the jump* of A. Note that $A <_T A'$ for all sets A. The jump operation can be iterated and so K'' denotes the *double jump* of the halting problem K.

Instead of considering computable learners, one can consider learners that are computable relative to some oracle. Our learning models so far feature a symmetry between the complexity of the learner and of the hypothesis space: the learner is a partial computable function and the hypotheses are indices for partial computable functions. One would accordingly allow an A-recursive learner to pick its hypotheses from an acceptable enumeration of the partial functions *computable in A*. What instead if we consider a learner that can access an oracle A but is asked to choose its conjectures from an enumeration of the partial computable functions? Relative to the complexity of the learner, the latter requirement can be seen as a limitation on the hypothesis space.

Our first result is a bit atypical, but fits the just described scenario. We consider a learner that is computable relative to an oracle for K' but is asked to use as hypothesis space an acceptable enumeration of programs not using any oracle. Theorem 11 below shows that the equivalence $\mathbf{Ex} = \mathbf{NUShEx}$ from [2] does *not* relativize to learners that are computable in K' but output codes for partial computable functions, and one can strengthen the separation to iterative learning. In the following result $\mathbf{It}[K']$ (respectively, $\mathbf{NUShEx}[K']$) denotes the collection of all classes of r.e. languages that are iteratively (respectively, non U-shapedly explanatory) learnable by some machine $\mathbf{M}$ that has access to the oracle K'. Such a machine (as in the definition of $\mathbf{Ex}$ and $\mathbf{It}$) outputs indices of partial computable functions and not of functions computed relative to K'.

Theorem 11. $\mathbf{It}[K'] \not\subseteq \mathbf{NUShEx}[K']$.

Proof. For every e, let $L_e = \{\langle e, x\rangle : x \in \mathbb{N}\}$ and $H_i = \{\langle e, x\rangle : x \leq \Phi_e^{K'}(e)\}$. Note that $\Phi_e^{K'}(e)$ is finite iff $e \in K''$. Now let

$$\mathcal{L} = \{L_e : e \in \mathbb{N}\} \cup \{H_e : e \in K''\}.$$

The class $\mathcal{L}$ is $\mathbf{It}[K']$-learnable. The learner outputs a hypothesis for the set given at the first case which applies:

- $\emptyset$ if no data of the form $\langle e, x\rangle$ has been seen so far;
- L_e if $x < \Phi_e^{K'}(e)$ for all data of the form $\langle e, x\rangle$ seen so far;
- H_e if $x \leq \Phi_e^{K'}(e)$ for all data of the form $\langle e, x\rangle$ seen so far and $x = \Phi_e^{K'}(e)$ for some datum;
- L_e if $x > \Phi_e^{K'}(e)$ for some datum of the form $\langle e, x\rangle$ seen so far.

It can easily be verified that the learner can keep track of the finitely many cases and update its hypothesis accordingly; within this process it uses two different indices for L_e in order to memorize whether a datum $\langle e, x\rangle$ with $x > \Phi_e^{K'}(e)$ has been seen so far or not.

Suppose by way of contradiction that $\mathbf{M}$ witnesses $\mathcal{L} \in \mathbf{NUShEx}[K']$. For every e, one can compute a number $f(e)$ such that $\mathbf{M}(\langle e, 0\rangle\langle e, 1\rangle \ldots \langle e, f(e)\rangle)$ outputs an index for L_e; this $f(e)$ must exist since $\mathbf{M}$ learns $\mathcal{L}$ and the number $f(e)$ can be found using the oracle K'. Since, $\mathbf{M}$ is not U-shaped, it implies that $\mathbf{M}$ does not change its mind on any text for a subset of L_e starting with $\langle e, 0\rangle\langle e, 1\rangle \ldots \langle e, f(e)\rangle$. Thus $f(e) > \Phi_e^{K'}(e)$ whenever $e \in K''$, so $K'' = \{e : \Phi_e^{K'}(e) \leq f(e)\}$ in contradiction to the fact that K'' is not K'-recursive. ∎

In the following, the above example is modified in order to carry over the separation to class-preserving learning (informally defined in Section 1.3). Lange and Zeugmann [26] introduced class-preserving learning and studied the dependency of learnability on the hypothesis space. We will introduce a bit of terminology (from [1]) to explain the notion. An infinite sequence $L_0, L_1, L_2, \ldots$ of recursive languages is called *uniformly recursive* if the set $\{\langle i, x\rangle : x \in L_i\}$ is recursive. A class $\mathcal{L}$ of recursive languages is said to be an *indexed family* of recursive languages if $\mathcal{L} = \{L_i : i \in \mathbb{N}\}$ for some uniformly recursive sequence $L_0, L_1, L_2, \ldots$; the latter is called a *recursive indexing* of $\mathcal{L}$.

Let $\mathcal{L}$ be an indexed family of recursive sets and let $H_0, H_1, H_2, \ldots$ be a hypothesis space, where one can recursively decide (in x and i) whether $x \in H_i$. We say that a machine $\mathbf{M}$ explanatorily identifies $\mathcal{L}$ *with respect to the hypothesis space* $H_0, H_1, H_2, \ldots$ iff for every $L \in \mathcal{L}$, on every text for L, $\mathbf{M}$ converges to some j such that $L = H_j$. A machine $\mathbf{M}$ *class-preserving* explanatorily identifies $\mathcal{L}$, if in the above situation $\{H_i : i \in \mathbb{N}\} = \mathcal{L}$. In what follows, for a learning criterion $\mathbf{I}$, $\mathbf{I}^{\mathrm{cp}}$ stands for class-preserving $\mathbf{I}$-learning, the collection of all classes of languages that can be $\mathbf{I}$-learned by some class-preserving machine.

Theorem 12. *There exists an indexed family in* $\mathbf{It}^{\mathrm{cp}} - \mathbf{NUShEx}^{\mathrm{cp}}$.

The positive side can be done using an indexed (recursive) family as hypothesis space, whereas the diagonalization against negative side can be done for any r.e. class preserving hypothesis space.

Proof. Fix an algorithmic enumeration $\mathbf{M}_0, \mathbf{M}_1, \ldots$ of learners [21]. Let $L_e = \{\langle e, x\rangle : x \in \mathbb{N}\}$ and let $L_e^n = \{\langle e, x\rangle : x < 2n \text{ or } x \text{ is odd}\}$. Let T_e denote a recursive text such that $T_e(x) = \langle e, x\rangle$. Let $S_e = \{\langle n, t\rangle : (\exists\, x \geq n)[\langle e, 2x\rangle \in W_{\mathbf{M}_e(T_e[2n]),t}]\}$. Now consider the class

$$\mathcal{L} = \{L_e : e \in \mathbb{N}\} \cup \{L_e^n : S_e \neq \emptyset \wedge \langle n, t\rangle = \min(S_e)\}.$$

The proof is now completed by showing the following two claims.

Claim 13. *$\mathcal{L} \notin$ **NUShEx**$^{\text{cp}}$.*

For proving Claim 13, suppose $\mathbf{M}_e$ witnesses $\mathcal{L} \in \mathbf{NUShEx}^{\text{cp}}$. Then, as $\mathbf{M}_e$ learns L_e, S_e is not empty. Let $\langle n, t\rangle$ be the least element of S_e. Now $W_{\mathbf{M}(T_e[2n])}$ must be a grammar for L_e (as no other language in $\mathcal{L}$ contains an element of form $\langle e, 2x\rangle$ for $x \geq n$). Let T be a text for L_e^n extending $T_e[2n]$. Now since $\mathbf{M}_e$ is non U-shaped on $\mathcal{L}$, $\mathbf{M}_e$, on T, does not abandon the hypothesis L_e since it is consistent with all upcoming data and is a language in $\mathcal{L}$. Thus $\mathbf{M}_e$ does not output any grammar for L_e^n beyond $T_e[2n]$. Thus, $\mathbf{M}_e$ does not learn L_e^n although $L_e^n \in \mathcal{L}$. This completes the proof of Claim 13.

Claim 14. *$\mathcal{L} \in$ **It**$^{\text{cp}}$.*

For proving Claim 14, let p be a 1–1 recursive function such that $p(e, 0)$ and $p(e, 1)$ are grammars/decision procedures for L_e, and $p(e, 2)$ is a grammar/decision procedure for L_e^n, if S_e is not empty and $\min(S_e) = \langle n, t\rangle$ for some t.

Now let $\mathbf{M}$ be an iterative learner which has the initial hypothesis ?, which keeps every hypothesis, including ?, on the datum # and which follows the following update procedure on a datum $\langle e, x\rangle$ where $a \in \{?, p(e, 0), p(e, 1), p(e, 2)\}$.

$$\mathbf{M}(a, \langle e, x\rangle) = \begin{cases} p(e, 0), & \text{if } a = \,?\, \text{ or } a = p(e, 0) \text{ and} \\ & S_e \text{ does not intersect } \{0, 1, \ldots, x\}; \\ p(e, 2), & \text{if } a = p(e, 0) \text{ and } S_e \text{ intersects } \{0, 1, \ldots, x\}; \\ p(e, 2), & \text{if } a = p(e, 2),\ \min(S_e) = \langle n, t\rangle \text{ and } \langle e, x\rangle \in L_e^n; \\ p(e, 1), & \text{otherwise.} \end{cases}$$

It is easy to verify that if S_e is empty, then $\mathbf{M}$ on any text for L_e outputs only $p(e, 0)$ as its conjecture (besides initial ?). If S_e is non-empty and $\min(S_e) = \langle n, t\rangle$, then for any text for L_e or L_e^n, $\mathbf{M}$ initially outputs ?, then outputs $p(e, 0)$, and eventually outputs $p(e, 2)$ (after seeing an input $\langle e, x\rangle$ such that $x \geq \langle n, t\rangle$). Beyond the first time $p(e, 2)$ is output, $\mathbf{M}$ changes its mind to $p(e, 1)$ iff it sees an input not contained in L_e^n. It follows that $\mathbf{M}$ learns $\mathcal{L}$. This completes the proof of Claim 14 and Theorem 12. ∎

5 Consistent and Monotonic Iterative Learning

Forbidding U-shapes is a *semantic* constraint on a learner's sequence of conjectures. In this section we study the interplay of this constraint with other well-studied semantic constraints, but in the memory-limited setting of iterative learning.

We now describe and then formally define the relevant variants of semantic constraints on the sequence of conjectures. Bārzdiņš [4] introduced *consistent learning* (in the context of function learning) where it is essentially required that the learner's conjectures do not contradict known data. Jantke [23] introduced *strong monotonic learning* which requires that every set generated by any new conjecture is a superset of the set generated by the previous one. Wiehagen [48] introduced the less-restrictive requirement of *monotonic learning* where, for each language L the learner actually learns, the intersection of L with the language generated by a learner's conjecture is a superset of the intersection of L with the language generated by any of the learner's previous conjectures.

Definition 15. [4, 23, 48] A learner $\mathbf{M}$ is *consistent* on a class $\mathcal{L}$ iff for all $L \in \mathcal{L}$ and all σ with $\mathrm{content}(\sigma) \subseteq L$, $\mathbf{M}(\sigma)$ is defined and $\mathrm{content}(\sigma) \subseteq W_{\mathbf{M}(\sigma)}$. **Cons** denotes the collection of all classes which have a **Ex**-learner which is consistent on the class of all sets. **ClassCons** denotes the collection of all classes $\mathcal{L}$ which have a **Ex**-learner which is consistent on $\mathcal{L}$.

A learner $\mathbf{M}$ is strong monotonic iff $W_i \subseteq W_j$ whenever $\mathbf{M}$ outputs, on any text for any language, at some time i and later j. **SMon** denotes the collection of all classes having a strong monotonic **Ex**-learner.

A learner $\mathbf{M}$ for $\mathcal{L}$ is monotonic iff $L \cap W_i \subseteq L \cap W_j$ whenever $\mathbf{M}$ outputs, on a text for some language $L \in \mathcal{L}$, at some time i and later j. **Mon** denotes the criterion of all classes having a monotonic **Ex**-learner.

Note that there are classes $\mathcal{L} \in$ **ClassCons** such that only partial learners witness this fact. Criteria can be combined. For example, **ItCons** is the criterion consisting of all classes which have an iterative and consistent learner. The indication of an oracle as in the criterion **ItConsSMon**$[K]$ below denotes that a learner for the given class must on the one hand be iterative, consistent and strong-monotonic while on the other hand the constraint of being recursive is weakened to the permission to access a halting-problem oracle for the inference process. Oracles can support the inference process, but there is no oracle permitting the learning of all classes (of r.e. languages) [22, 34]. The next result gives some basic connections between iterative, strongly monotonic and consistent learning.

Theorem 16. (a) **ItCons** $\subseteq$ **ItConsSMon**.
(b) **ConsSMon** $\subseteq$ **ItConsSMon**.
(c) **SMon** $\subseteq$ **ItConsSMon**$[K]$.

Proof. (a) Given an iterative consistent learner $\mathbf{M}$ for $\mathcal{L}$, let — as in the case of normal learners — $\mathbf{M}(\sigma)$ denote the hypothesis which $\mathbf{M}$ makes after having seen the sequence σ. Now define a recursive one-one function f such that, for every index e, $W_{f(e)} = \bigcup_{\sigma \in \{\sigma' : \mathbf{M}(\sigma') = e\}} \mathrm{content}(\sigma)$. Since $\mathbf{M}$ is consistent, $\mathrm{content}(\sigma) \subseteq W_{\mathbf{M}(\sigma)}$ for all σ and so $W_{f(e)} \subseteq W_e$. The new learner $\mathbf{N}$ is the modification of $\mathbf{M}$ which outputs $f(e)$ instead of e; $\mathbf{N}$ is consistent since whenever one can reach a hypothesis e through a string containing a datum x then $x \in W_{f(e)}$. Since f is one-one, $\mathbf{N}$ is also iterative and follows the update rule $\mathbf{N}(f(e), x) = f(\mathbf{M}(e, x))$.

It is easy to see that $\mathbf{N}$ is strongly monotonic: Assume that $\mathbf{M}(e, y) = e'$ and x is any element of $W_{f(e)}$. Then there is a σ with $\mathbf{M}(\sigma) = e$ and $x \in \mathrm{content}(\sigma)$. It follows that $\mathbf{M}(\sigma y) = e'$, $x \in \mathrm{content}(\sigma y)$ and $x \in W_{f(e')}$. So $W_{f(e)} \subseteq W_{f(e')}$ and the transitiveness of the inclusion gives the strong monotonicity of $\mathbf{N}$.

It remains to show that $\mathbf{N}$ learns $\mathcal{L}$. Let $L \in \mathcal{L}$ and T be a text for L and e be the index to which $\mathbf{M}$ converges on T. The learner $\mathbf{N}$ converges on T to $f(e)$. Since $W_e = L$ it holds that

$W_{f(e)} \subseteq L$. Furthermore, for every n there is $m > n$ with $\mathbf{M}(T[m]) = e$, thus $T(n) \in W_{f(e)}$ and $L \subseteq W_{f(e)}$. This completes the proof of part (a).

(b) A consistent learner never outputs ?. Now, given a strong monotonic and consistent learner $\mathbf{M}$ for some class $\mathcal{L}$, one defines a recursive one-one function $f : (\mathbb{N} \cup \{\#\})^* \to \mathbb{N}$ such that

$$W_{f(\sigma)} = W_{\mathbf{M}(\sigma)} \cup \mathrm{content}(\sigma)$$

and initializes a new iterative learner $\mathbf{N}$ with the hypothesis $f(\lambda)$ and the following update rule for the hypothesis $f(\sigma)$ and observed datum x:

- If $\mathbf{M}(\sigma x) = \mathbf{M}(\sigma)$ then $\mathbf{N}(f(\sigma), x) = f(\sigma)$;
- If $\mathbf{M}(\sigma x) \neq \mathbf{M}(\sigma)$ then one takes the length-lexicographic first extension τ of σx such that $W_{\mathbf{M}(\eta),|\sigma|} \subseteq \mathrm{content}(\tau)$ for all $\eta \preceq \sigma$ and defines $\mathbf{N}(f(\sigma), x) = f(\tau)$.

Note that in the second case, $\mathrm{content}(\tau) = \mathrm{content}(\sigma x) \cup (\bigcup_{\eta \preceq \sigma} W_{\mathbf{M}(\eta),|\sigma|})$ and that the length-lexicographic ordering is just taken to single out the first string with this property with respect to some ordering. The new iterative learner is strongly monotonic since whenever it changes the hypothesis then it does so from $f(\sigma)$ to $f(\tau)$ for some τ extending σ and thus $W_{f(\sigma)} = \mathrm{content}(\sigma) \cup W_{\mathbf{M}(\sigma)} \subseteq \mathrm{content}(\tau) \cup W_{\mathbf{M}(\tau)} = W_{f(\tau)}$ (as $\mathbf{M}$ is strong monotonic). Furthermore, $\mathbf{N}$ is also consistent: whenever it sees a number x outside $W_{f(\sigma)}$ then x is also outside $W_{\mathbf{M}(\sigma)}$ and $\mathbf{M}(\sigma x) \neq \mathbf{M}(\sigma)$ by the consistency of $\mathbf{M}$. Then the new τ constructed contains x explicitly and therefore $x \in W_{\mathbf{N}(f(\sigma),x)}$. By the strong monotonicity of $\mathbf{N}$, an element once incorporated into a hypothesis is also contained in all future hypotheses. So it remains so show that $\mathbf{N}$ actually learns $\mathcal{L}$.

Given $L \in \mathcal{L}$ and a text T for L, there is a sequence of strings $\sigma_0, \sigma_1, \ldots$ such that $\sigma_0 = \lambda$ and $\mathbf{N}(f(\sigma_n), T(n)) = f(\sigma_{n+1})$. By induction one can show that $\sigma_n \in (L \cup \{\#\})^*$ and $W_{\mathbf{M}(\sigma_n)} \subseteq L$ for all n. There are two cases.

First, there is an n such that $\sigma_m = \sigma_n$ for all $m \geq n$. Then $L \subseteq W_{f(\sigma_n)}$ since $\mathbf{N}$ is a consistent learner and eventually converges to this hypothesis on the text L. Furthermore, $W_{f(\sigma_n)} \subseteq L$ as mentioned above, so $\mathbf{N}$ learns L.

Second, for every n there is an $m > n$ such that σ_m is a proper extension of σ_n. Let T' be the limit of all σ_n. One can easily see that T' contains data from two sources, some items taken over from T and some elements taken from sets $W_{\mathbf{M}(\eta)}$ with $\eta \preceq \sigma_n$ for some n; since $\mathbf{M}$ is strong monotonic these elements are all contained in L and so $\mathrm{content}(T') \subseteq L$. Furthermore, for every n the element $T(n)$ is contained in $W_{f(\sigma_{n+1})}$ and thus there is an extension σ_k of σ_{n+1} which is so long that

$$T(n) \in W_{\mathbf{M}(\sigma_{n+1}),|\sigma_k|} \cup \mathrm{content}(\sigma_{n+1}).$$

If then for some $m \geq k$ the string σ_{m+1} is a proper extension of σ_m, then $T(n) \in \mathrm{content}(\sigma_{m+1})$. As a consequence, T' is a text for L on which $\mathbf{M}$ converges to a hypothesis e. Then, one has that for all sufficiently large m, where σ_{m+1} is a proper extension of σ_m, σ_{m+1} is actually an extension of $\sigma_m T(m)$ and $\mathbf{M}(\sigma_m T(m)) = \mathbf{M}(\sigma_m)$, which would by construction enforce that $\mathbf{N}$ does not update its hypothesis and $\sigma_{m+1} = \sigma_m$. By this contradiction, the second case does not hold and the first applies, thus $\mathbf{M}$ learns $\mathcal{L}$. This completes the proof of part (b).

(c) This case is parallel to (b), but here the learner $\mathbf{M}$ is not consistent. Therefore one uses the oracle K to check for consistency and has the following modified update algorithm for $\mathbf{N}$; this test cannot be done without an oracle.

- If $\mathbf{M}(\sigma x) = \mathbf{M}(\sigma)$ and $x \in W_{f(\sigma)}$ then $\mathbf{N}(f(\sigma), x) = f(\sigma)$;
- If $\mathbf{M}(\sigma x) \neq \mathbf{M}(\sigma)$ or $x \notin W_{f(\sigma)}$ then one takes the length-lexicographic first extension τ of σx such that $W_{\mathbf{M}(\eta),|\sigma|} \subseteq \text{content}(\tau)$ for all $\eta \preceq \sigma$ and defines $\mathbf{N}(f(\sigma), x) = f(\tau)$.

The verification is the same as part (b), except that the consistency of **N** follows now directly from the explicit test and the strong monotonicity of **N**. So part (d) follows. ∎

One might ask whether the halting problem oracle is necessary in inclusion (d). The following example gives an affirmative answer where K is again the diagonal halting problem.

Example 17. $\mathbf{SMon} \subseteq \mathbf{It}[A]$ *only if* $K \leq_T A$.

Proof. Let H be an infinite recursive set disjoint to K. Such a set exists since K is not simple [31]. Now define the class $\mathcal{L}$ to consist of the set K and all sets of the form $K \cup D$ where D is finite, nonempty and has a maximum in H.

A strongly monotonic learner for $\mathcal{L}$ with input $T[n]$ first computes E_n as being the content of $T[n]$, then computes $S_n = \{x \in E_n : \exists y \geq x\, (y \in E_n \cap H)\}$ and finally conjectures $K \cup S_n$. To achieve syntactic convergence, a hypothesis is only updated when $n = 0$ or $n > 0 \wedge S_n \neq S_{n-1}$.

Note that whenever $L \in \mathcal{L}$ then the set $S = \{x \in L : \exists y \geq x\, (y \in L \cap H)\}$ is finite and equal to S_n for almost all n. Thus the learner converges to an index for $K \cup S$ and this convergence is syntactic as $S_n = S_{n-1} = S$ for almost all n. So the given learner is a **Ex**-learner. Since $S_0 \subseteq S_1 \subseteq S_2 \subseteq \ldots$, the learner is also strongly monotonic. This completes the proof for $\mathcal{L} \in \mathbf{SMon}$.

Now consider an iterative learner **M** for $\mathcal{L}$ with oracle A. This learner **M** has a locking sequence for K and converges on this locking sequence to a hypothesis e. For all $x \in K$ it holds that $\mathbf{M}(e, x) = e$ by the locking sequence property. For all $x \notin K$ it holds that $\mathbf{M}(e, x) \neq e$ since otherwise **M** cannot distinguish a text for $K \cup \{y\}$ from a text for $K \cup \{x, y\}$ whenever $y > x \wedge y \in H$; such an y exists since H is infinite. It follows that $K = \{x : \mathbf{M}(e, x) = e\}$ and thus $K \leq_T A$. This completes the proof. ∎

Note that the proof of Theorem 16 (a) needs that the learner is an **ItCons**-learner and not just an **ItClassCons**-learner. In the latter case, the inference process cannot be enforced to be strong-monotonic as the following example shows.

Example 18. *The class* $\mathcal{L}$ *containing the set* $\{0, 2, 4, 6, 8, \ldots\}$ *of even numbers and all sets* $\{0, 2, 4, \ldots, 2n\} \cup \{2n + 1\}$ *with* $n \in \mathbb{N}$ *is in* $\mathbf{ItClassCons} - \mathbf{SMon}$.

Proof. On one hand, the learner which conjectures the set of even numbers until an element of the form $2n + 1$ is seen and then changes to the unique possible hypothesis $\{0, 2, 4, \ldots, 2n\} \cup \{2n + 1\}$ is easily seen to be class-consistent and iterative. So $\mathcal{L} \in \mathbf{ItClassCons}$.

On the other hand, a given learner for $\mathcal{L}$ has eventually to conjecture an index for $\{0, 2, 4, \ldots\}$ after having seen enough even numbers. Let n be larger than any number seen by the learner before the conjecture is made as above. Then, the input text might actually be for the language $\{0, 2, 4, \ldots, 2n\} \cup \{2n + 1\}$: in which case the learner would be forced to change its mind non-strong monotonically. Hence, $\mathcal{L} \notin \mathbf{SMon}$. ∎

So, class-consistent iterative learners cannot be made strong monotonic, even with an oracle. However, the next result shows that they can still be made monotonic, *and*, simultaneously, non U-shaped.

Theorem 19. *If $\mathcal{L} \in$ **ItClassCons** then $\mathcal{L}$ has a monotonic non U-shaped iterative learner.*

Proof. Suppose **M ItClassCons**-identifies $\mathcal{L}$. We write $\mathbf{M}(x_1, x_2, \ldots, x_r)$ for the hypothesis obtained by feeding $x_1, x_2, \ldots, x_r$ one after the other into the learner; this notion has the initial hypothesis for $r = 0$. We say $(x_1, x_2, \ldots, x_r)$ (here r maybe 0) is valid if for all $i < r$ (including $i = 0$), $\mathbf{M}(x_1, \ldots, x_i) \downarrow \neq \mathbf{M}(x_1, \ldots, x_i, x_{i+1}) \downarrow$. For valid $(x_1, x_2, \ldots, x_r)$ and $k \leq r$, we define the set

$$W_{F(k,x_1,x_2,\ldots,x_r)} = \{x_i : 1 \leq i \leq k\} \cup \{x : (\exists s \leq k)[\mathbf{M}(x_1, \ldots, x_s, x) = \mathbf{M}(x_1, \ldots, x_s) \text{ and } (\forall w : s \leq w \leq r)[x \in W_{\mathbf{M}(x_0,\ldots,x_w)}]]\}$$

and point out that the next two claims follow immediately from the definition of F:

Claim 20. *Suppose $(x_1, \ldots, x_{r+1})$ is valid and $k \leq r$. Then, $W_{F(k,x_1,x_2,\ldots,x_r)} \supseteq W_{F(k,x_1,x_2,\ldots,x_r,x_{r+1})}$.*

Claim 21. *Suppose $(x_1, \ldots, x_r)$ is valid and $k \leq k' \leq r$. Then, $W_{F(k,x_1,\ldots,x_r)} \subseteq W_{F(k',x_1,\ldots,x_r)}$.*

Furthermore, in the next claim, (a), (b) follow from definition of F and consistency of $\mathbf{M}$ on L and (c) follows from (b) and definition of F.

Claim 22. *Suppose $(x_1, \ldots, x_{r+1})$ is valid and $k \leq r$. Further suppose $\{x_1, \ldots, x_{r+1}\} \subseteq L$ and $L \in \mathcal{L}$. Then,*

(a) $W_{F(k,x_1,\ldots,x_r)} \subseteq W_{\mathbf{M}(x_1,\ldots,x_r)}$;
(b) $W_{F(k,x_1,\ldots,x_r)} \cap L \subseteq W_{\mathbf{M}(x_1,\ldots,x_r,x_{r+1})}$;
(c) $W_{F(k,x_1,\ldots,x_r)} \cap L \subseteq W_{F(k,x_1,\ldots,x_r,x_{r+1})}$.

Claim 23. *Suppose $(x_1, \ldots, x_{r'})$ is valid and $k \leq k'$ and $r \leq r'$. Further suppose $\{x_1, \ldots, x_{r'}\} \subseteq L$ and $L \in \mathcal{L}$. Then, $W_{F(k,x_1,\ldots,x_r)} \cap L \subseteq W_{F(k',x_1,\ldots,x_{r'})}$.*

This claim needs a short proof: $W_{F(k,x_1,\ldots,x_r)} \cap L \subseteq W_{F(k,x_1,\ldots,x_{r'})}$, follows from Claim 22 (c), $W_{F(k,x_1,\ldots,x_{r'})} \subseteq W_{F(k',x_1,\ldots,x_{r'})}$, follows from Claim 21. Thus Claim follows.

Now we continue with the proof of the main result and define $\mathbf{N}$ as follows. Suppose on input text seen so far, $\mathbf{M}$ has made mind changes on $x_1, \ldots, x_r$, and k is the smallest number such that, for all x seen in input so far, there exists $s \leq k$, such that $\mathbf{M}(x_1, \ldots, x_s) = \mathbf{M}(x_1, \ldots, x_s, x)$ (note that, by induction, such k can be iteratively found and will be $\leq r$). $\mathbf{N}$ then outputs $F(k, x_1, \ldots, x_r)$.

Claim 23 implies $\mathbf{N}$ is monotonic for the class $\mathcal{L}$.

Now consider any text T for $L \in \mathcal{L}$. $\mathbf{N}$ converges on T as $\mathbf{M}$ converges on T. Suppose $\mathbf{N}$ on T converges to $F(k, x_1, \ldots, x_r)$. Then, by Claim 22(a) and $L \in \mathbf{Ex}(\mathbf{M})$, we immediately have that $W_{F(k,x_1,\ldots,x_r)} \subseteq L$. Furthermore, $L \subseteq W_{F(k,x_1,\ldots,x_r)}$, as $\mathbf{M}$ is iterative, and for every $x \in \text{content}(T)$ there exists a $s \leq k$ such that $\mathbf{M}(x_1, \ldots, x_s, x) = \mathbf{M}(x_1, \ldots, x_s)$. Thus, $\mathbf{N}$ learns L in the limit.

To see non U-shaped learning of $\mathbf{N}$, consider $L \in \mathcal{L}$ and suppose on text T, at some point $\mathbf{N}$ outputs, $F(k, x_1, \ldots, x_r)$ and this enumerates exactly L. This implies by definition of F that, for all $x \in L$, either x equals $x_1, \ldots, x_k$ or there exists an $s \leq k$ such that $\mathbf{M}(x_1, \ldots, x_s) = \mathbf{M}(x_1, \ldots, x_s, x)$. It follows by definition of $\mathbf{N}$ that from then on $\mathbf{N}$ only outputs grammars of form $F(k, x_1, \ldots, x_r, \ldots, x_{r'})$. But then Claim 20 and Claim 22(c) imply that $F(k, x_1, \ldots, x_r, \ldots, x_{r'})$ is also a grammar for L. ∎

6 Memoryless Feedback Learning

An iterative learner has a severe memory limitation: it can store no previously seen data. On the other hand, crucially, an iterative learner remembers its previous conjecture. In this section we introduce a model of learning in which the learner does *not* remember its last conjecture *and* can store no previous input data. The learner is instead allowed to make, at each stage of its learning process, n feedback queries asking whether some n data items have been previously seen. We call such learners *n-memoryless feedback learners*, and the main result of the present section (Theorem 30) shows that U-shaped behaviour is necessary for the full learning power of n-memoryless feedback learning. At the end of the present section (Theorem 32) we prove that, as might be expected, being able to do $n+1$ feedback queries gives more learning power than being able to do only n.

We now proceed with the formal definition of n-memoryless feedback learning.

Definition 24. Suppose $n \geq 0$. An *n-memoryless feedback learner* $\mathbf{M}$ has as input one datum from a text. It then can make n-queries which are calculated from its input datum. These queries are as to whether some n data items were already seen previously in the text. From its input and the answers to these queries, it either outputs a conjecture or the ? symbol. That is, given a language L and a text T for L, $\mathbf{M}(T(k))$ is determined as follows: First, n-values $q_i(T(k)), i = 1, \ldots, n$, are computed. Second, n bits $b_i, i = 1, \ldots, n$ are determined and passed on to $\mathbf{M}$, where each b_i is 1 if $q_i(T(k)) \in \mathrm{content}(T[k])$ and 0 otherwise. Third, an hypothesis e_k is computed from $T(k)$ and the b_i's. $\mathbf{M}$ $\mathbf{MLF}_n$-learns L if, for all T for L, for $\mathbf{M}$ on T, there is an k such that $W_{e_k} = L$ and $e_m \in \{?, e_k\}$ for all $m > k$. $\mathbf{MLF}_n$ denotes the class of all classes learnable by a n-memoryless feedback learner.

In what follows D_i is the finite set with canonical index i [37]: i algorithmically codes both the cardinality of D_i and how to decide membership in D_i.

Remark 25. One can generalize $\mathbf{MLF}_n$ to $\mathbf{MLF}_*$. Each $\mathbf{MLF}_*$-learner employs a recursive function F mapping $\mathbb{N}$ to finite subsets of $\mathbb{N}$ such that, for every x, the learner asks whether any of the $y \in D_{F(x)}$ have been seen before. Depending on the answers, the learner outputs a hypothesis or ?. Clearly, by Theorem 32, $\mathbf{MLF}_*$ is a proper superset of $\mathbf{MLF}_n$.

On one hand, $\mathbf{It} \not\subseteq \mathbf{MLF}_*$ since the class of all sets $\{0, x\}$ and $\{1, x\}$ with $x \in \{2, 3, 4, \ldots\}$ is not learnable by an $\mathbf{MLF}_*$ learner. For $y = 0$ or 1, if the learner makes a conjecture on input y, where all the questions are answered no, then it is wrong on input xy^∞ for some $x \notin D_{F(y)}$. On the other hand, if the learner does not make a conjecture on both the inputs 0 and 1 where all questions are answered "no", then it clearly does not identify one of the sets $\{0, x\}$ or $\{1, x\}$ for some $x \notin D_{F(0)} \cup D_{F(1)}$.

On the other hand, let $E = \{2x : x \in \mathbb{N}\}$, $A_p = E \cup \{p^i : i \in \mathbb{N}\}$ and $B_p = E \cup \{p^i : i \in \mathbb{N}\} - \{2p\}$. Then, $\mathcal{L} = \{E\} \cup \{A_p : p \text{ is odd prime}\} \cup \{B_p : p \text{ is odd prime}\}$ is not iteratively learnable. But one can verify that it is in $\mathbf{MLF}_1$: Let x be the input, let p be an odd prime and let i be a positive integer; if $x \in E$ and 1 has not been seen so far (as verified by a query) then the learner conjectures E; if $x = p^i$ and $2p$ has been seen so far then the learner conjectures A_p; if $x = p^i$ and $2p$ has not been seen so far then the learner conjectures B_p; in all other cases the learner abstains from a new conjecture.

The next result shows that non U-shaped 1-memoryless feedback learners are strictly less powerful than unrestricted 1-memoryless feedback learners: There exists a class of languages that can be learned by a 1-memoryless feedback learner only if the learner is allowed to make some U-shapes on some text for some language in the class. The basic idea for the proof is to include in the class two types of sets that start differing after a non-computable point. After this proof we indicate how to adapt it to show that U-shaped learning is necessary at *each* level of the $\mathbf{MLF}_n$-hierarchy (see Theorem 30 below).

Theorem 26. $\mathbf{NUShMLF}_1 \subset \mathbf{MLF}_1$.

Proof. The idea is to use, for every e, two sets L_e, H_e such that the learner can easily figure out that it has to learn one of these sets but is nevertheless forced to oscillate between these two hypotheses and is therefore U-shaped. These two sets are equal up to some value $F(e)$, where

$$F(e) = \max(\{1 + \varphi_i(e) : i \leq e \text{ and } \varphi_i(e)\downarrow\} \cup \{0\}).$$

Note that F grows faster than any partial or total recursive function. Based on this function F one now defines the family $\mathcal{L} = \{L_0, L_1, L_2, \ldots\} \cup \{H_0, H_1, H_2, \ldots\}$ where

$$L_e = \{\langle e, x\rangle : x < F(e) \text{ or } x \text{ is even}\};$$
$$H_e = \{\langle e, x\rangle : x < F(e) \text{ or } x \text{ is odd}\}.$$

We first show that $\mathcal{L} \in \mathbf{MLF}_1$. Note that the learning algorithm cannot store the last guess due to its memory limitation but might output a '?' in order to repeat that hypothesis. The parameter e is visible from each current input except '#'. The algorithm is the following:

If the new input is # or if the input is $\langle e, x\rangle$ and the Feedback says that $\langle e, x+1\rangle$ has already appeared in the input earlier, then output ?. Otherwise, if input is $\langle e, x\rangle$ and $\langle e, x+1\rangle$ has not yet appeared in input, then output a canonical grammar for L_e (H_e) if x is even (odd).

Consider any text T for L_e. Let n be such that content$(T[n]) \supseteq L_e \cap \{\langle e, x\rangle : x \leq F(e)+1\}$. Then, it is easy to verify that, the learner will either output ? or a conjecture for L_e beyond $T[n]$. On the other hand, for any even $x > F(e)$, if $T(m) = \langle e, x\rangle$, then the learner outputs a conjecture for L_e after having seen $T[m+1]$ (this happens infinitely often, by definition of L_e). Thus, the learner $\mathbf{MLF}_1$-identifies L_e. Similar argument applies for H_e.

We now show that $\mathcal{L} \notin \mathbf{NUShMLF}_1$. So suppose by way of contradiction that the learner $\mathbf{M}$ $\mathbf{NUShMLF}_1$-identifies $\mathcal{L}$. We now do the following analysis.

We assume without loss of generality that $\mathbf{M}$'s query on input $\langle e, x\rangle$ is of form $\langle e, x'\rangle$ for some x'. If $\mathbf{M}(\langle e, x\rangle)$ makes the query $\langle e, x'\rangle$, then we let $Q(\langle e, x\rangle) = x'$.

Claim 27. (a) *There do not exist infinitely many e such that for some x, $\mathbf{M}(\langle e, x\rangle)$ outputs a hypothesis on yes answer to feedback query.*

(b) *There do not exist infinitely many e such that for some x, $\mathbf{M}(\langle e, x\rangle)$ does not pose a query, but outputs an hypothesis.*

Of this claim, we show part (a). Part (b) can be shown similarly. Suppose by way of contradiction otherwise. Define partial function η to be $\eta(e) = \max(\{x_e, Q(\langle e, x_e\rangle)\})$, where x_e is the first number found, if any, such that $\mathbf{M}(\langle e, x_e\rangle)$ on answer yes to query, outputs a hypothesis. Now $F(e) > \eta(e)$ (if $\eta(e)$ is defined), for all but finitely many e. Thus, we have that for infinitely many e, $x_e < F(e), Q(\langle e, x\rangle) < F(e)$, and on answer yes, $\mathbf{M}(\langle e, x_e\rangle)$ outputs a hypothesis. Pick any such e. Without loss of generality assume that $\mathbf{M}(\langle e, x_e\rangle)$ is not a grammar for L_e (case of H_e is similar). Consider the text T for L_e which starts with $\langle e, Q(\langle e, x_e\rangle)\rangle$ and has $\langle e, x_e\rangle$ in every alternate position of the input. Now $\mathbf{M}$ on T infinitely often outputs a hypothesis which is not for L_e (whenever it sees $\langle e, x_e \rangle$ in the input), and thus $\mathbf{M}$ does not **Ex**-identify L_e. This completes the proof of the claim.

Now we continue with the main proof. As finitely many L_e/H_e can easily be learned, for the following analysis, we may assume without loss of generality that for any input, $\mathbf{M}$ outputs an hypothesis only when making a query and getting a no answer. We further assume without loss of generality that if $\mathbf{M}$ does not output a hypothesis on no-query, then it does not make the query at all (since the query in this case is not used). Thus, all and only the no-answer queries lead to hypothesis output by $\mathbf{M}$.

Claim 28. *For any e, if $\{\langle e, x\rangle, \langle e, x'\rangle\} \subseteq L_e$, and $\mathbf{M}(\langle e, x\rangle)$ on no answer to query, outputs a grammar for L_e, and $\mathbf{M}(\langle e, x'\rangle)$ on no answer to query, outputs a grammar which is not for L_e, then $Q(\langle e, x'\rangle) = x$. Similar result holds when L_e above is replaced by H_e.*

For a proof of this claim, assume that it is wrong and consider the text T for L_e starting with $\langle e, x\rangle\langle e, x'\rangle$. $\mathbf{M}$ is non U-shaped on T.

Claim 29. (a) *There exist only finitely many e such that* $\text{card}(\{Q(\langle e, x\rangle) : x \in \mathbb{N}\}) \geq 3$.
(b) *There exist only finitely many e such that* $\text{card}(\{Q(\langle e, x\rangle) : x \in \mathbb{N}\}) = 1$.
(c) *There exist only finitely many e such that* $\text{card}(\{Q(\langle e, x\rangle) : x \in \mathbb{N}\}) = 2$.

The main result is now obtained by proving this claim.

(a) Suppose by way of contradiction otherwise. Let η be a partial function such that, $\eta(e) = \max(\{x_1^e, x_2^e, x_3^e\})$, where $Q(\langle e, x_1^e\rangle)$, $Q(\langle e, x_2^e\rangle)$ and $Q(\langle e, x_3^e\rangle)$, are all different (if there are no such x_1^e, x_2^e, x_3^e, then $\eta(e)$ is undefined). Now by definition of F, for all but finitely many e, $F(e) > \eta(e)$, if it is defined. Pick any e in domain of η such that $F(e) > \eta(e)$. Note that $\langle e, x_1^e\rangle, \langle e, x_2^e\rangle, \langle e, x_3^e\rangle$ are in both L_e and H_e. Let x_L^e, x_H^e be such that $\langle e, x_L^e\rangle \in L_e$ and $\langle e, x_H^e\rangle \in H_e$ and $\mathbf{M}(\langle e, x_L^e\rangle)$ outputs a grammar for L_e on answer no to query and $\mathbf{M}(\langle e, x_H^e\rangle)$ outputs a grammar for H_e on answer no to query (note that there exist such x_L^e, x_H^e, since otherwise $\mathbf{M}$ does not **Ex**-identify L_e, H_e). Let x_e^j, $j \in \{1, 2, 3\}$ be such that $Q(\langle e, x_e^j\rangle) \notin \{x_L^e, x_H^e\}$. Without loss of generality suppose that $\mathbf{M}(\langle e, x_e^j\rangle)$ is not a grammar for L_e (case of H_e is similar). Then, on any text T for L_e starting with $\langle e, x_L^e\rangle\langle e, x_e^j\rangle$, $\mathbf{M}$ is not U-shaped.

(b) Suppose by way of contradiction otherwise. Let η be a partial function such that $\eta(e) = Q(\langle e, x\rangle)$, for the first x found such that $\mathbf{M}(\langle e, x\rangle)$ asks a query (and outputs an hypothesis on no-answer). Pick an e such that $\text{card}(\{Q(\langle e, x\rangle) : x \in \mathbb{N}\}) = 1$ and $\eta(e)\downarrow < F(e)$ (all but finitely many e such that $\text{card}(\{Q(\langle e, x\rangle) : x \in \mathbb{N}\}) = 1$, satisfy this condition). Let q_e be the only member of $\{Q(\langle e, x\rangle) : x \in \mathbb{N}\}$. Note that q_e belongs to both L_e and H_e. But then, for any text for L_e or H_e which starts with q_e, $\mathbf{M}$ does not make any further hypothesis beyond $T[1]$. Thus, $\mathbf{M}$ cannot **Ex**-identify both L_e and H_e.

(c) Similar to part (b), by using η to bound the two potential queries, and starting the text with both these queries. This completes the proof of the claim.

Now, parts (a)—(c) of Claim 29 immediately lead to a contradiction. ∎

It is not difficult to see that the proof of Theorem 26 can be adapted to show there is a class in $\mathbf{MLF}_n$ that is not $\mathbf{MLF}_n$-learnable without U-shapes. This can be achieved by adding $n-1$

special elements, $s_1, s_2, \ldots, s_{n-1}$, to the languages used in Theorem 26. Then, the machine from the proof of the positive part of Theorem 26 can be modified as follows. If it sees these special elements, the learner outputs ?. If the learner sees any other element x, then the learner queries whether $s_1, \ldots, s_{n-1}$ are in the input besides the main query and outputs conjectures as before, with the special elements answered yes being added to the conjecture. The negative direction of the proof can proceed essentially as before, as the last conjectures of the learner needs to correctly determine whether the special elements are in the input or not. We omit the details. This gives us the following Theorem, showing that U-shaped learning *is* necessary for full learning power of n-memoryless feedback learners, for all $n > 0$. For $n = 0$, $\mathbf{NUShMLF}_0 = \mathbf{MLF}_0$, see Remark 38 in Section 7 below.

Theorem 30. *For all $n > 0$, $\mathbf{NUShMLF}_n \subset \mathbf{MLF}_n$.*

It is reasonable to ask, as we do in the Conclusion (Section 8), whether the need for U-shaped learning in Theorem 26 can be removed by allowing more queries. That is, can we show that there are $\mathbf{MLF}_1$-learnable classes that, for all $n > 1$, are not $\mathbf{MLF}_n$-learnable *without* U-shapes. The class from Theorem 26 is perhaps such an example although it is $\mathbf{NUShMLF}_*$-learnable.

Finally, an iterative *total* learner that can store one selected previous datum is called a $\mathbf{Bem}_1$-learner (1-bounded example memory learner) in [13, 34]. One can also consider a "memoryless" version of this concept, where a learner does not memorize its previous hypothesis, but, instead, memorizes one selected previous datum. Under both these criteria, the class $\{\{0, x\}, \{1, x\} : x > 1\}$ from Remark 25 is non U-shapedly learnable with the corresponding memory bound; indeed it would even be finitely learnable if one allows memorization of data, without outputing a hypothesis. But this class is not in $\mathbf{MLF}_*$ as mentioned above. So we have the following proposition.

Proposition 31. $\mathbf{NUShBem}_1 \not\subseteq \mathbf{NUShMLF}_*$.

To conclude this section we show that the n-memoryless feedback criteria form a hierarchy of more and more powerful learning criteria increasing in the number n of feedback queries allowed.

Theorem 32. *For all $n > 0$, $\mathbf{NUShMLF}_{n+1} \not\subseteq \mathbf{MLF}_n$.*

Proof. Fix an algorithmic enumeration $\mathbf{M}_0, \mathbf{M}_1, \ldots$ of learners [21]. We diagonalize against this enumeration. Let $L_i = \{\langle 0, x\rangle : x \leq n\} \cup \{\langle i+1, x\rangle : x \in \mathbb{N}\}$. Let $L_i^S = S \cup \{\langle i+1, 2x\rangle : x \in \mathbb{N}\}$. $\mathcal{L}$ will contain L_i, and maybe $L_i^{S_i}$, for some i, *where* S_i is defined just below.

S_i is defined as follows (using some standard search): search for $y \in L_i - \{\langle 0, x\rangle : x \in \mathbb{N}\}$ such that $\mathbf{M}_i(y)$ queries $q_1, ..., q_n$ and outputs a grammar which contains at least $n+2$ elements outside $\{\langle i+1, 2x\rangle : x \in \mathbb{N}\}$, where the answers given to $q_1, \ldots, q_n$ are 'yes' if and only if they belong to L_i. Then, $S_i = \{y, q_1, \ldots, q_n\} \cap L_i$, where S_i is defined based on the first success found in the above search in some standard method of searching.

Claim: $\mathcal{L}$ is not in $\mathbf{MLF}_n$. Suppose by way of contradiction that some learner $\mathbf{MLF}_n$-learns $\mathcal{L}$. Take i so large that $\mathbf{M}_i$ is equivalent to the given learner and L_i is not among the sets conjectured by the learner on any σ with $\mathrm{content}(\sigma) \subseteq \{\langle 0, x\rangle : x \leq n\}$ and $|\sigma| \leq n+1$.

Now, if the search for S_i does not succeed then $\mathbf{M}_i$ does not $\mathbf{MLF}_n$-identify L_i. Otherwise, $\mathbf{M}_i$ does not $\mathbf{MLF}_n$-identify $L_i^{S_i}$.

Claim: $\mathcal{L}$ is in $\mathbf{NUShMLF}_{n+1}$. On inputs of form $\langle i+1, 2x\rangle$ query elements $\langle 0, x\rangle$ such that $x \leq n$. If all are present, then conjecture L_i. Otherwise search for S_i as above for x steps. If found, then conjecture, $L_i^{S_i}$. Otherwise conjecture ?. It is easy to verify the claim. ∎

7 Bounded Memory States Learning

Memoryless feedback learners store no information about the past. Bounded memory states learners, introduced in this section, have no memory of previous conjectures but can store a bounded number of values in their long term memory. This model allows one to separate the issue of a learner's ability to remember its previous conjecture from the issue of a learner's ability to store information about the previously seen input. Similar models of machines with bounded long term memory are studied in [24]. We now proceed with the formal definition.

Definition 33. [24] For $c > 0$, a c-bounded memory states learner is a (possibly partial) function

$$\mathbf{M} : \{0, 1, \dots, c-1\} \times (\mathbb{N} \cup \#) \to (\mathbb{N} \cup \{?\}) \times \{0, 1, \dots, c-1\}$$

which maps the old long term memory content plus a datum to the current hypothesis plus the new long term memory content. The long term memory has the initial value 0. There is no initial hypothesis.

$\mathbf{M}$ learns a class $\mathcal{L}$ iff, for every $L \in \mathcal{L}$ and every text T for L, there is a sequence $a_0, a_1, \dots$ of long term memory contents and $e_0, e_1, \dots$ of hypotheses and a number n such that, for all m, $a_0 = 0$, $W_{e_n} = L$, $\mathbf{M}(a_m, T(m)) = (e_m, a_{m+1})$ and $m \geq n \Rightarrow e_m \in \{?, e_n\}$. We denote by $\mathbf{BMS}_c$ the collection of classes learnable by a c-bounded memory states learner.

The next result shows that for bounded memory states learning, the concepts of explanatory and behaviourally correct learning essentially coincide.

Theorem 34. *If* $\mathbf{M}$ *has a constant bound* $c > 0$ *on its long term memory and identifies a class* $\mathcal{L}$ *in behaviourally correct way, then there is a further learner with memory bound* $(c+1)!$ *which identifies* $\mathcal{L}$ *with at most* $2 \cdot c$ *mind changes.*

Proof. The proof follows ideas outlined in [24]. The idea is to build a new learner $\mathbf{N}$ which simulates $\mathbf{M}$ but which assumes certain old data-items to be (virtually) repeated while processing the text in order to undo certain changes of state or hypotheses. The learner $\mathbf{N}$ cannot access these data-items explicitly but reconstruct from the data seen so far whether it should copy a hypothesis of $\mathbf{M}$ or replace it by the symbol ? and whether it should go into a new state or not. In order to do this, the long term memory of $\mathbf{N}$ stores the following pieces of information:

- A sequence $q_1, q_2, \dots, q_n$ of long term memory contents visited so far by $\mathbf{M}$;
- The index n of the last element q_n;
- The index m of the last q_m such that $\mathbf{N}$ has output some hypothesis when going from q_m either to itself or to q_{m+1}; $m = 0$ if $\mathbf{N}$ has not output any hypothesis.

Note that n can take values from 1 to c, m can take values from 0 to c and that the sequence $q_1, q_2, \dots, q_n$ is the initial part of some permutation of the c elements in the set $\{0, 1, \dots, c-1\}$. So one can extend this to a full permutation by assigning arbitrary values to the remaining elements. Furthermore, $q_1 = 0$ and $q_k > 0$ for $k > 1$, thus there are $(c-1)!$ many possible values for the sequence $q_1, q_2, \dots, q_n$. In addition one has c many possible values for n and $c+1$ many possible values for m, giving in total $(c+1)!$ possible values for the long term memory.

Now the new learner $\mathbf{N}$ starts with the long term memory content such that $n = 1, q_1 = 0, m = 0$. The update rule is the following for a data-item x, where $e \in \mathbb{N}$ (that is, $e \neq ?$) in the case distinction:

(1) if $\mathbf{M}(q_n, x) = (?, r)$ for some $r \in \{q_1, \ldots, q_n\}$ then $\mathbf{N}$ does not change its long term memory and conjectures ?;
(2) if $\mathbf{M}(q_n, x) = (?, r)$ for some $r \notin \{q_1, \ldots, q_n\}$ then $\mathbf{N}$ defines $q_{n+1} = r$, updates $n = n + 1$ and conjectures ?;
(3) if $\mathbf{M}(q_n, x) = (e, q_k)$ for some $k \in \{1, \ldots, m\}$ then $\mathbf{N}$ does not change its long term memory and conjectures ?;
(4) if $\mathbf{M}(q_n, x) = (e, q_k)$ for some $k \in \{m+1, \ldots, n\}$ then $\mathbf{N}$ updates $m = n$ and conjectures e;
(5) if $\mathbf{M}(q_n, x) = (e, r)$ for some $r \notin \{q_1, \ldots, q_n\}$ then $\mathbf{N}$ defines $q_{n+1} = r$, updates $m = n$, updates $n = n + 1$ and conjectures e.

For the verification let a language $L \in \mathcal{L}$ and a text T for L be given. The underlying idea is the following: the learner $\mathbf{N}$ always assumes that it "virtually inserts" so many data that the simulated machine $\mathbf{M}$ has after seeing the real datum x and the virtually repeated data the long term memory q_n. In Case (1), $\mathbf{M}$ assumes that the data which let $\mathbf{M}$ go from r to q_n is repeated such that either only ? are output or the last hypothesis e' of $\mathbf{N}$ shows up as the last hypothesis generated by this sequence. In Case (3), $\mathbf{N}$ knows that it goes to a long term memory q_k from which one can go with some data through q_m to a point where $\mathbf{N}$'s last hypothesis e' was output and then go to q_n with $\mathbf{M}$ only outputting no-conjecture-symbols. Thus $\mathbf{N}$ virtually inserts these data and omits the hypothesis e as it would be overwritten by e'. In Cases (4) and (5), $\mathbf{N}$ copies the hypothesis e as it would overwrite the last hypothesis e'. Note that if $m = 0$ the learner goes automatically through one of these two cases whenever $\mathbf{M}(q_n, x)$ causes a hypothesis. The remaining part of the verification is that the ideas of virtually inserting data would transform a given text T to a text T' such that either $\mathbf{M}$ does not make a hypothesis on T' at all and only outputs no-conjecture-symbols or that $\mathbf{M}$ outputs finitely many hypotheses with the last one being the same as the last hypothesis of $\mathbf{N}$ or that $\mathbf{M}$ outputs infinitely many hypotheses with the last one of $\mathbf{N}$ occurring infinitely often in the sequence. Since $\mathbf{M}$ has also to learn L from T', one can conclude that $\mathbf{N}$ outputs some hypotheses and that its last hypothesis e is correct since it is either also the last hypothesis of $\mathbf{M}$ on T' or $\mathbf{M}$ outputs e infinitely often on T'.

To see the mind change bound, one has only to look at how many hypotheses are output while n has a fixed value k. These are at most two hypotheses, at the first, m is updated from some value below k to k, at the second, a new element is added to the list and n is updated from k to $k + 1$. This completes the proof. ∎

We now state and prove the main result of the present section, showing that every 2-bounded memory states learner *can* be simulated by a non U-shaped one. Note that $\mathbf{NUShBMS}_1 = \mathbf{BMS}_1$ by Remark 38 below in this section.

Theorem 35. $\mathbf{BMS}_2 \subseteq \mathbf{NUShBMS}_2$.

Proof. Let $\mathcal{L}$ be given and assume that $\mathbf{M}$ witnesses $\mathcal{L} \in \mathbf{BMS}_2$. We assume without loss of generality that $\mathbf{M}$ does not change its memory on input #, as otherwise we could easily modify $\mathbf{M}$ to work without any memory.

In the following, "$*$" stands for the case that the value does not matter and in all (legal) cases the same is done.

Define a function P such that $P(?) =?$ and, for $e \in \mathbb{N}$, $P(e)$ is an index of the set $W_{P(e)} = \bigcup_{s \in S(e)} W_{e,s}$ where $S(e)$ is the set of all s satisfying either (a) or ((b) and (c) and (d)) below:

(a) There exists an $x \in W_{e,s}$, $\mathbf{M}(1, x) = (*, 0)$;

(b) For all $x \in W_{e,s}$, $[\mathbf{M}(0,x) = (*,1) \Rightarrow \mathbf{M}(1,x) = (?,1)]$;
(c) There exists an $x \in W_{e,s}$, $\mathbf{M}(0,x) = (?,1)$ or for all $x \in W_{e,s}$, $\mathbf{M}(0,x) = (*,0)$;
(d) For all $x \in W_{e,s} \cup \{\#\}$, $[\mathbf{M}(0,x) = (j,*) \Rightarrow W_{e,s} \subseteq W_j \ \wedge\ W_{j,s} \subseteq W_e]$.

Now we define for all $m \in \{0,1\}$, $j \in \mathbb{N} \cup \{?\}$ and $x \in \mathbb{N} \cup \{\#\}$,

$$\mathbf{N}(m,x) = \begin{cases} (P(j),0), & \text{if } m = 0 \text{ and } \mathbf{M}(0,x) = (j,0); \\ (j,1), & \text{if } m = 0 \text{ and } \mathbf{M}(0,x) = (*,1) \text{ and } \mathbf{M}(1,x) = (j,*) \text{ and } j \neq ?; \\ (j,1), & \text{if } m = 0 \text{ and } \mathbf{M}(0,x) = (j,1) \text{ and } \mathbf{M}(1,x) = (?,*); \\ (j,1), & \text{if } m = 1 \text{ and } \mathbf{M}(1,x) = (j,*). \end{cases}$$

Now fix an $L \in \mathcal{L}$ and text T for L. Note that $\mathbf{M}$ identifies L. We show below that $\mathbf{N}$ will also identify L from text T.

Case (1): For all $x \in L$, $\mathbf{M}(0,x) = (*,0)$.
Then $\mathbf{N}$ behaves exactly like $\mathbf{M}$ with the only difference that every hypothesis e is translated to $P(e)$. As $\mathbf{M}$ converges syntactically to a hypothesis e, $\mathbf{N}$ converges syntactically to the hypothesis $P(e)$. One can now verify that every s goes into $S(e)$ by satisfying the conditions (b), (c) and (d) and thus $W_{P(e)} = W_e$: (b) and (c) are clearly satisfied; for condition (d) note that all hypotheses output by $\mathbf{M}$ are e since otherwise $\mathbf{M}$ would diverge on fat texts for L, that is, on texts where every datum of $L \cup \{\#\}$ appears infinitely often.

Case (2): Not Case (1) and there exists an $x \in L$, $\mathbf{M}(1,x) = (*,0)$.
First we show that the learner $\mathbf{N}$ outputs at least one conjecture on T. Assume by way of contradiction that $\mathbf{N}$ on T for L does not output any hypothesis. We then show that there is a text T' for L on which $\mathbf{M}$ does not output any hypothesis. Let m be the first number such that after seeing $T[m]$, $\mathbf{M}$ has memory 1. Then $\mathbf{N}$ and $\mathbf{M}$ both do not output any hypothesis on this initial portion $T[m]$. T' will be a modification of T by inserting appropriate elements in order to force $\mathbf{M}$ back to memory 1 without outputting a hypothesis. Let T' be the limit of σ_n defined as follows. One starts with $\sigma_0 = T[m]$ and now defines σ_{n+1} inductively from σ_n: if $\mathbf{M}$ after $\sigma_n T(m+n)$ has the memory 1 we just set $\sigma_{n+1} = \sigma_n T(m+n)$ else we set $\sigma_{n+1} = \sigma_n T(m+n) T[m]$ in order to transfer $\mathbf{M}$ back into memory 1 without outputting a conjecture. Now T' is the limit of all σ_n and $\mathbf{M}$ does not output any hypothesis on T'. But T' is just T with $T[m]$ inserted at some places and therefore T' is a text for L. Hence $\mathbf{M}$ does not identify L, a contradiction. So $\mathbf{N}$ does output a conjecture on T.

Similarly, we can show that $\mathbf{M}$ outputs only one hypothesis e on data coming from L since otherwise one could create a text T' on which $\mathbf{M}$ outputs infinitely often two different hypotheses. So $\mathbf{N}$ makes at most one mind change, potentially from $P(e)$ to e, and thus $\mathbf{N}$ is non U-shaped. For correctness, note that $W_{P(e)} = W_e$ since (a) holds.

Case (3): Not Cases (1) and (2) and there exists an $x \in L$ such that $\mathbf{M}(0,x) = (*,1)$ and $\mathbf{M}(1,x) \neq (?,1)$.
In this case, all conjectures of $\mathbf{N}$ before getting memory 1 are wrong. This is because, for any e, $W_{P(e)}$ either contains an x such that $\mathbf{M}(1,x) = (*,0)$ (due to condition (a)), or it does not contain any x such that $\mathbf{M}(0,x) = (*,1)$ and $\mathbf{M}(1,x) \neq (?,1)$ (due to condition (b)). Also, once $\mathbf{N}$ has memory content 1, it will only output correct grammars, since $\mathbf{M}$ outputs only correct grammars after having memory 1 — otherwise $\mathbf{M}$ would output infinitely often wrong grammars on a fat text for L. The output during transition from memory 0 to 1 thus does not

effect U-shapedness. $\mathbf{N}$ does $\mathbf{Ex}$-identify L, as it will output a correct grammar for L once it sees the input x.

Furthermore, $\mathbf{N}$ converges on T since $\mathbf{M}$ converges on $T(0)T(0)T(1)T(1)T(2)T(2)\ldots$ which is obviously also a text for L.

Case (4): Not Cases (1), (2), (3) and for all $x \in L$, $\mathbf{M}(0, x) \neq (?, 1)$.

In this case, $\mathbf{N}$ does change memory, outputs a grammar at the point of changing memory and then follows $\mathbf{M}$. Thus it $\mathbf{Ex}$-identifies L. We now claim that every grammar output by $\mathbf{N}$ before it changes memory to 1 is incorrect. Suppose by way of contradiction that $P(e)$ output by $\mathbf{N}$ before it changes memory to 1 is a grammar for L. By hypothesis of current case, there is an $x \in L$ such that $\mathbf{M}(0, x) = (*, 1)$. Fix one such x. Now, for all s with $W_{e,s} \subseteq L$, if $x \in W_{e,s}$, the conditions (a) and (c) do not hold and $s \notin S(e)$. Thus, either $W_{P(e)}$ is not a subset of L or $W_{P(e)}$ does not contain x, a contradiction. Hence $\mathbf{N}$ is non U-shaped.

Case (5): Not Cases (1), (2), (3), (4). That is, the following three conditions hold:

- for all $x \in L$, $\mathbf{M}(1, x) = (*, 1)$;
- for all $x \in L$, if $\mathbf{M}(0, x) = (*, 1)$ then $\mathbf{M}(1, x) = (?, 1)$;
- there exists an $x \in L$ such that $\mathbf{M}(0, x) = (?, 1)$.

Note that $\mathbf{M}$ necessarily outputs correct hypotheses after it achieves memory 1 (since otherwise $\mathbf{M}$ would output infinitely often wrong grammars on a fat text for L).

Subcase (5–I): L contains only finitely many elements x such that $\mathbf{M}(1, x) = (j, 1)$, $j \neq ?$.

We first claim that for all $j \neq ?$, such that $\mathbf{M}(0, x) = (j, *)$, $W_j = L$. Suppose otherwise. Let y be such that $\mathbf{M}(0, y)$ outputs a wrong hypothesis. Let $X = \{x \in L : \mathbf{M}(1, x) = (j, 1), j \neq ?\}$. Note that, for all $x \in X$, we must have $\mathbf{M}(0, x) = (j, 0)$ for some $j \in \mathbb{N} \cup \{?\}$, by the hypothesis of the current case. Let $z \in L$ be such that $\mathbf{M}(0, z) = (?, 1)$ (there exists such a z by hypothesis of current case). Let σ be such that content$(\sigma) = X$. let T'' be a text for $L - X$. Now consider the text $T' = \sigma y T''$ if $\mathbf{M}(0, y) = (*, 1)$ and $T' = \sigma y z T''$ otherwise. Then, $\mathbf{M}$ on T' never outputs a conjecture beyond σy, and thus it converges on T' to a wrong hypothesis for L, a contradiction. It follows that $\mathbf{M}$ always outputs correct hypothesis (or ?) on input from $L \cup \{\#\}$ (for any memory value). Thus, all hypothesis output by $\mathbf{N}$ are also correct (since conditions (b), (c), (d) hold for large enough s, in the definition of $S(e)$, for any e such that $W_e = L$). Hence, $\mathbf{N}$ $\mathbf{NUShEx}$-identifies L.

Subcase (5–II): L contains infinitely many elements x such that $\mathbf{M}(1, x) = (j, 1)$, $j \neq ?$.

In this case $\mathbf{N}$ clearly outputs a conjecture after achieving memory 1 and thus $\mathbf{N}$ converges on T to the same hypothesis as $\mathbf{M}$ on T. So $\mathbf{N}$ $\mathbf{Ex}$-identifies L from T.

Now, if for all $x \in L \cup \{\#\}$, $\mathbf{M}(*, x)$ output a correct hypothesis, if any, then $W_{P(e)} = W_e = L$ for all hypothesis e output by $\mathbf{M}$ on input x (for any memory value), since conditions (b), (c) and (d) hold for large enough s in the definition of $S(e)$. Thus, $\mathbf{N}$ only outputs correct hypotheses and $\mathbf{N}$ is non U-shaped.

On the other hand, if there exists an $x \in L \cup \{\#\}$ such that $\mathbf{M}(0, x)$ outputs a wrong hypothesis. Then all grammars output by $\mathbf{N}$ before changing memory to 1 are not for L. This holds as for any e, if $W_e \neq L$, then $W_{P(e)}$ is either finite or equal to W_e, and hence not equal to L. On the other hand if $W_e = L$, then in the computation of $W_{P(e)}$, (a) does not hold, and (d) can hold only for finitely many s, and hence $W_{P(e)} \neq L$. Hence $\mathbf{N}$ is non U-shaped on L. ▮

In the just previous proof, the modification of W_e to $W_{P(e)}$ is essential. If this were not be permitted and we considered class-preserving learners only, the result changes, as the following remark shows.

Remark 36. The proof of Theorem 12 above provides a class $\mathcal{L} \in (\mathbf{It}^{\text{cp}} - \mathbf{NUShEx}^{\text{cp}})$, where the superscript cp stands for class-preserving learning.

This same $\mathcal{L}$ is also in $\mathbf{BMS}_2^{\text{cp}}$ as the learner $\mathbf{M}$ from the proof of Claim 14 can be modified to obtain the machine $\mathbf{M}'$ witnessing this fact. Recall that

$$\mathcal{L} = \{L_e : e \in \mathbb{N}\} \cup \{L_e^n : S_e \neq \emptyset \wedge \langle n, t\rangle = \min(S_e)\},$$

where S_e was a certain set defined in dependence of the e-th learner from an enumeration of all learners. Furthermore, recall that $p(e, 0)$ generates L_e and $p(e, 2)$ generates L_e^n, where p is from the proof of Claim 14. $\mathbf{M}'$ starts with long term memory 0 and works on input $\langle e, x\rangle$ as follows:

$$\mathbf{M}'(a, \langle e, x\rangle) = \begin{cases} (p(e,0), 0) & \text{if } a = 0 \text{ and } S_e \text{ does not contain any } \langle n, t\rangle \leq x; \\ (p(e,2), 1) & \text{if } a = 0 \text{ and } S_e \text{ contains an element } \langle n, t\rangle \leq x; \\ (p(e,0), 1) & \text{if } a = 1 \text{ and } S_e \text{ contains the} \\ & \text{(least) element } \langle n, t\rangle \leq x, \text{ and } \langle e, x\rangle \notin L_n^e; \\ (?, 1) & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}'$ **Ex**-identifies the class $\mathcal{L}$ — employing long term memory $\{0, 1\}$.

To conclude the present section we state the following Theorem that the c-bounded memory state criteria form a hierarchy of more and more powerful learning criteria increasing in the number c of states allowed. Note that 1-bounded memory state learners can identify singleton classes consisting of one set. The class

$$\mathcal{L}_c = \{\{0, 1, 2\}, \{0, 1, 2, 3\}, \{0, 1, 2, 3, 4\}, \ldots, \{0, 1, 2, \ldots, 2c\}\}$$

from [24, discussion after Theorem 7.6] witnesses the properness of the following inclusion. The discussion there can be easily extended to show $\mathcal{L}_c$ is not in $\mathbf{BMS}_{c-1}$. We give the proof for completeness.

Theorem 37. *For all $c > 1$, $\mathbf{BMS}_{c-1} \subset \mathbf{BMS}_c$.*

Proof. Consider the class

$$\mathcal{L}_c = \{\{0, 1, 2\}, \{0, 1, 2, 3\}, \{0, 1, 2, 3, 4\}, \ldots, \{0, 1, 2, \ldots, 2c\}\}$$

as in [24, Discussion after Theorem 7.6]. In [24] it is shown that this class is learnable with c long term memory states but not learnable with less than $2c-2$ mind changes. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{BMS}_{c-1}$-learns $\mathcal{L}_c$. Define σ_i, for $i \leq 2c-2$, such that content$(\sigma_i) = \{x : x \leq i+2\}$, and $\sigma_i \subseteq \sigma_{i+1}$, and $\mathbf{M}(\sigma_i)$ is a grammar for $\{x : x \leq i+2\}$. Let the states of $\mathbf{M}$ after receiving σ_i be a_i. We claim that a_{2j}, $j \leq c-1$ must all be pairwise distinct, and hence $\mathbf{M}$ uses at least c memory values. If not, then suppose $a_j = a_{j+k}$, where $j+k \leq c-1$. Let τ be such that $\sigma_j\tau = \sigma_{j+k}$. Note that τ is non-empty, and $\mathbf{M}$ makes at least 2 distinct conjectures between σ_j (exclusive) and σ_{j+k} (inclusive). Thus, $\mathbf{M}$ on $\sigma_j\tau^\infty$ makes infinitely many mind changes, even though content$(\sigma_j\tau) \in \mathcal{L}_c$. Theorem follows. ∎

Remark 38. One can generalize $\mathbf{BMS}_c$ to **ClassBMS** and **BMS**. The learners for these criteria use natural numbers as long term memory. For **ClassBMS** we have the additional constraint that for every text of a language inside the *learnt* class, there is a constant c depending on the text such that the value of the long term memory is never a number larger than c. For **BMS** the corresponding constraint applies to all texts for all sets, even those outside the class.

One can extend methods used above for $\mathbf{BMS}_c$ and results from [24] to prove that $\mathbf{ClassBMS} = \mathbf{It}$. Furthermore, a class is in **BMS** iff it has a confident iterative learner, that is, an iterative learner which converges on every text, whether this text is for a language in the class to be learned or not.

It is easy to see that $\bigcup_{c\in\mathbb{N}} \mathbf{BMS}_c \subset \mathbf{BMS} \subset \mathbf{ClassBMS}$. Furthermore, one has by Remark 25 that there is a class in $\mathbf{MLF}_1 - \mathbf{ClassBMS}$. The bottom levels of the hierarchies coincide: $\mathbf{BMS}_1 = \mathbf{MLF}_0$. These levels are nontrivial as they already contain every uniformly recursive class of *disjoint* languages; the disjointness is important, since, for $a \neq b$, an $\mathbf{MLF}_0$-learner cannot learn the class of languages $\{\{a\}, \{b\}, \{a, b\}\}$. It is easy to argue that $\mathbf{MLF}_0 = \mathbf{NUShMLF}_0$.

Furthermore, there is a class in $\mathbf{BMS}_2 - \mathbf{MLF}_*$. To see this let $L_i = \{\langle i+1, x\rangle : x \in \mathbb{N}\}$ and $L_{i,x} = L_i \cup \{\langle 0, \langle i, x\rangle\rangle\}$. Let the class be $\{L_i : i \in \mathbb{N}\} \cup \{L_{i,x} : i, x \in \mathbb{N}\}$.

8 Conclusions and Open Problems

Numerous results related to non U-shaped learning for machines with severe memory limitations were obtained. In particular, it was shown that

- there are class-preservingly iteratively learnable classes that cannot be learned without U-shapes by any iterative class-preserving learner (Theorem 12),
- class-consistent iterative learners for a class can be turned into iterative non U-shaped *and* monotonic learners for that class (Theorem 19),
- for all $n > 0$, there are n-memoryless feedback learnable classes that cannot be learned without U-shapes by any n-memoryless feedback learner (Theorem 30) and, by contrast,
- every class learnable by a 2-bounded memory states learner can be learned by a 2-bounded memory states learner without U-shapes (Theorem 35).

The above results are, in our opinion, interesting in that they show how the impact of forbidding U-shaped learning in the context of severely memory-limited models of learning is far from trivial. In particular, the tradeoffs that our results reveal between remembering one's previous conjecture, having a long-term memory, and being able to make feedback queries are delicate and perhaps surprising. Many fascinating problems remain open.

For memoryless feedback learning, it is open whether the following strengthening of Theorem 30 is true.

Problem 39. *Is* $\mathbf{MLF}_1 \subseteq \mathbf{NUShMLF}_n$*, for* $n > 1$*?*

Finally, for state bounded memory learning, it is open whether our Theorem 35 generalizes to the case of learners that are allowed to store one among c values for $c > 2$.

Problem 40. *Is* $\mathbf{BMS}_c \subseteq \mathbf{NUShBMS}_c$*, for* $c > 2$*?*

Also, the question of the necessity of U-shaped behaviour with respect to the stronger memory-limited variants of **Ex**-learning (bounded example memory and feedback learning) from the previous literature [27, 13] remains wide open. Humans can remember *much* more than one bit and likely retain something of their prior hypotheses; furthermore, they have some access to knowledge of whether they've seen something before. Hence, the open problems of this section may prove interesting for cognitive science.

9 Acknowledgement

We thank the anonymous referees of COLT for several helpful comments.

References

1. Dana Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
2. Ganesh Baliga, John Case, Wolfgang Merkle, Frank Stephan and Rolf Wiehagen. *When unlearning helps.* Technical Report TRA5/06, School of Computing, National University of Singapore, 2005. Preliminary version appeared as "Unlearning helps" by G. Baliga, J. Case, W. Merkle and F. Stephan in *Proceedings of the 27th International Colloquium on Automata, Languages and Programming* – ICALP 2000, Springer Lecture Notes in Computer Science 1853:844–855, 2000.
3. Janis Bārzdiņš. Two theorems on the limiting synthesis of functions. In *Theory of Algorithms and Programs, volume 1*, pages 82–88. Latvian State University, 1974. In Russian.
4. Janis Bārzdiņš. Inductive Inference of automata, functions and programs. *International Mathematical Congress, Vancouver*, pages 771–776, 1974.
5. Lenore Blum and Manuel Blum. Towards a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.
6. Manuel Blum. A machine independent theory of the complexity of the recursive functions. *Journal of the Association for Computing Machinery* 14:322–336, 1967.
7. T. G. R. Bower. Concepts of development. In *Proceedings of the 21st International Congress of Psychology*. Presses Universitaires de France, pages 79–97, 1978.
8. Melissa Bowerman. Starting to talk worse: Clues to language acquisition from children's late speech errors. In S. Strauss and R. Stavy, editors, *U-Shaped Behavioral Growth*. Academic Press, New York, 1982.
9. Susan Carey. Face perception: Anomalies of development. In S. Strauss and R. Stavy, editors, *U-Shaped Behavioral Growth*, Developmental Psychology Series. Academic Press, pages 169–190, 1982.
10. Lorenzo Carlucci, John Case, Sanjay Jain and Frank Stephan. U-shaped learning may be necessary, *Journal of Computer and Systems Sciences*, *to appear*.
11. Lorenzo Carlucci, Sanjay Jain, Efim Kinber and Frank Stephan. Variations on U-shaped learning. *Eighteenth Annual Conference on Learning Theory*, Colt 2005, Bertinoro, Italy, June 27–30, 2005, Proceedings. Lecture Notes in Computer Science, 3559:382–397, 2005.
12. John Case. The power of vacillation in language learning. *SIAM Journal on Computing*, 28(6):1941–1969, 1999.
13. John Case, Sanjay Jain, Steffen Lange and Thomas Zeugmann. Incremental Concept Learning for Bounded Data Mining. *Information and Computation*, 152(1):74–110, 1999.

14. John Case and Chris 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*, Lecture Notes in Computer Science, 140:107–115, 1982.
15. John Case and Carl H. Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25:193–220, 1983.
16. Rusins Freivalds, Efim B. Kinber and Carl H. Smith. On the impact of forgetting on learning machines. *Journal of the ACM*, 42:1146–1168, 1995.
17. Rusins Freivalds, Efim B. Kinber and Carl H. Smith. Probabilistic versus Deterministic Memory Limited Learning. *Algorithmic Learning for Knowledge-Based Systems, GOSLER Final Report*, Lecture Notes in Computer Science, 961:155–161, 1995.
18. Mark Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.
19. Mark Fulk, Sanjay Jain and Daniel Osherson. Open problems in "Systems That Learn". *Journal of Computer and System Sciences*, 49:589–604, 1994.
20. E. Mark Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
21. Sanjay Jain, Daniel Osherson, James Royer and Arun Sharma. *Systems that Learn: An Introduction to Learning Theory.* MIT Press, Cambridge, Mass., second edition, 1999.
22. Sanjay Jain and Arun Sharma. On the non-existence of maximal inference degrees for language identification. *Information Processing Letters*, 47:81–88, 1993.
23. Klaus-Peter Jantke. Monotonic and non-monotonic Inductive Inference, *New Generation Computing*, 8:349–360, 1991.
24. Efim Kinber and Frank Stephan. Language learning from texts: mind changes, limited memory and monotonicity. *Information and Computation*, 123:224–241, 1995.
25. David Kirsh. PDP learnability and innate knowledge of language. In S. Davis, editor, *Connectionism: Theory and Practice*, pages 297–322. Oxford University Press, 1992.
26. Steffen Lange and Thomas Zeugmann. The learnability of recursive languages in dependence on the space of hypotheses. *GOSLER-Report*, 20/93. Fachbereich Mathematik und Informatik, TH Leipzig, 1993.
27. Steffen Lange and Thomas Zeugmann. Incremental Learning from Positive Data. *Journal of Computer and System Sciences*, 53:88–103, 1996.
28. Nancy Ann Lynch, Albert R. Meyer and Michael J. Fischer. Relativization of the Theory of Computational Complexity. *Transactions of the American Mathematical Society*, 220:243–287, 1976.
29. Gary Marcus, Steven Pinker, Michael Ullman, Michelle Hollander, T. John Rosen and Fei Xu. *Overregularization in Language Acquisition.* Monographs of the Society for Research in Child Development, volume 57, no. 4. University of Chicago Press, 1992. Includes commentary by Harold Clahsen.
30. Samuel Moelius. Private Communication, 2006.
31. Piergiorgio Odifreddi. *Classical Recursion Theory.* North Holland, Amsterdam, 1989.
32. Piergiorgio Odifreddi. *Classical Recursion Theory*, volume II. Elsevier, Amsterdam, 1999.
33. Daniel Osherson and Scott Weinstein. Criteria of language learning. *Information and Control*, 52:123–138, 1982.
34. Daniel Osherson, Michael Stob and Scott Weinstein. *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists.* MIT Press, 1986.

35. Steven Pinker. Formal models of language learning. *Cognition*, 7:217–283, 1979.
36. Kim Plunkett and Virginia Marchman. U-shaped learning and frequency effects in a multi-layered perceptron: implications for child language acquisition. *Cognition*, 38(1):43–102, 1991.
37. Hartley Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York, 1967. Reprinted, MIT Press, 1987.
38. James Royer. *A Connotational Theory of Program Structure*. Lecture Notes in Computer Science, 273. Springer, 1987.
39. James Royer and John Case. *Subrecursive Programming Systems: Complexity and Succinctness*. Research monograph in *Progress in Theoretical Computer Science*. Birkhäuser Boston, 1994.
40. Carl H. Smith. The power of pluralism for automatic program synthesis. *Journal of the Association of Computing Machinery*, 29:1144–1165, 1982.
41. Robert Soare. *Recursively Enumerable Sets and Degrees*. Springer-Verlag, 1987.
42. Sidney Strauss and Ruth Stavy, editors. *U-Shaped Behavioral Growth*. Developmental Psychology Series. Academic Press, 1982.
43. Sidney Strauss, Ruth Stavy and N. Orpaz. The child's development of the concept of temperature. Manuscript, Tel-Aviv University. 1977.
44. Niels A. Taatgen and John R. Anderson. Why do children learn to say broke? A model of learning the past tense without feedback. *Cognition*, 86(2):123–155, 2002.
45. Kenneth Wexler. On extensional learnability. *Cognition*, 11:89–95, 1982.
46. Kenneth Wexler and Peter W. Culicover. *Formal Principles of Language Acquisition*. MIT Press, 1980.
47. Rolf Wiehagen. Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. *Journal of Information Processing and Cybernetics*, 12:93–99, 1976.
48. Rolf Wiehagen. A thesis in Inductive Inference. *Nonmonotonic and Inductive Logic, First International Workshop*, Lecture Notes in Artificial Intelligence, 543:184–207, 1990.