Solutions to Open Questions for Non-U-Shaped Learning with Memory Limitations

John Case[1] and Timo Kötzing[2,⋆]

[1] Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716-2586, USA
case@cis.udel.edu
[2] Max-Planck-Institut für Informatik, Campus E 1 4, 66123 Saarbrücken, Germany
koetzing@mpi-inf.mpg.de

Abstract. A U-shape occurs when a learner first learns, then unlearns, and, finally, relearns, some target concept. Within the framework of Inductive Inference, previous results have shown, for example, that U-shapes are unnecessary for explanatory learning, but are necessary for behaviorally correct learning.

This paper solves the following two problems regarding non-U-shaped learning posed in the prior literature.

First, it is shown that there are classes learnable with three memory states that are not learnable non-U-shapedly with any finite number of memory states. This result is surprising, as for learning with one or two memory states, U-shapes are known to be unnecessary.

Secondly, it is shown that there is a class learnable memorylessly with a single feedback query such that this class is not learnable non-U-shapedly memorylessly with any finite number of feedback queries. This result is complemented by the result that any class of *infinite* languages learnable memorylessly with finitely many feedback queries is so learnable without U-shapes – in fact, all classes of infinite languages learnable with *complete* memory are learnable memorylessly with finitely many feedback queries and without U-shapes. On the other hand, we show that there is a class of infinite languages learnable memorylessly with a single feedback query, which is *not* learnable with*out* U-shapes by any particular bounded number of feedback queries.

Keywords: Inductive Inference.

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.

⋆ Timo Kötzing was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant no. NE 1182/5-1.

M. Hutter et al. (Eds.): ALT 2010, LNAI 6331, pp. 285–299, 2010.

1.1 U-Shaped Learning

U-shaped learning occurs when a learner first learns a correct behavior, then abandons that correct behavior 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 [SS82], understanding of temperature [SS82], weight conservation [SS82], object permanence [SS82] and face recognition [Car82]. The case of language 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 behavior has figured prominently in cognitive science [MPU$^+$92, TA02].

While the prior cognitive science literature on U-shaped learning was typically concerned with modeling *how* humans achieve U-shaped behavior, [BCM$^+$08, CCJS07a] are motivated by the question of *why* humans exhibit this seemingly inefficient behavior. 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 behavior is logically necessary? We first need to describe some formal criteria of successful learning.

An algorithmic learning function h is, in effect, fed an infinite sequence consisting of the elements of a (formal) language L in arbitrary order with possibly some pause symbols # in between elements. During this process the machine outputs a corresponding sequence $p(0), p(1), \ldots$ of hypotheses (grammars) which may generate the language L to be learned. A fundamental criterion of successful learning of a language is called *explanatory learning* (TxtEx-*learning*) and was introduced by Gold [Gol67]. 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. Formally, *behaviorally correct learning* [CL82, OW82] 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* [Cas99, JORS99]. This paradigm involves learning criteria which allow the learner to vacillate in the limit between *at most* some finite number of 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.

[BCM$^+$08] showed that every TxtEx-learnable class of languages is TxtEx-learnable by a non U-shaped learner, that is, for TxtEx-learnability, U-shaped learning is *not* necessary. Furthermore, based on a proof in [FJO94], [BCM$^+$08]

noted that, by contrast, for behaviorally correct learning [CL82, OW82], U-shaped learning *is* necessary for full learning power. In [CCJS07a] it is shown that, for non-trivial vacillatory learning, U-shaped learning is again necessary (for full learning power). Thus, in many contexts, seemingly inefficient U-shaped learning can actually increase one's learning power.

1.2 Memory-Limited Learning

It is clear that human learning involves memory limitations. In the present paper (as in [CCJS07b]) we consider the necessity of U-shaped learning in some formally *memory-limited* versions of language learning. In the prior literature many types of memory-limited learning have been studied [LZ96, WC80, Wie76, OSW86, FJO94, CJLZ99, CCJS07b]. Herein we study the types from [CCJS07b] about which that paper has some results, and answer the open questions from that paper about those types.

1.3 Brief Summary of Main Results

The paper [CCJS07b] introduces *Bounded Memory State (BMS) learning*. Associated learners do *not* have access to any previously seen data. Instead, after each datum presented, the learner may choose one out of a bounded number of memory states, and, when presented with another datum, will be passed this memory state along with the new datum. Thus, each output of new conjecture and new memory state may depend only on the new datum and the just previous memory state. Intuitively, such a learner can be pictured as a finite state machine, where the transitions depend on each new datum.[1]

In [CCJS07b], the authors show that Bounded Memory States learning with up to two memory states does not require U-shapes.

As an open problem (Problem 40) they ask whether or not U-shapes are similarly unnecessary for higher numbers of memory states. Surprisingly, Theorem 3 says that there is a class learnable with *three* memory states which is not learnable for *any* number of memory states and with*out* U-shapes. Hence, in all but the bottom two cases for number of memory states available, *U-shapes are necessary* for increased learning power.

Also in [CCJS07b], *Memoryless Feedback (MLF) learning* is introduced. This is similar to BMS learning in that a learner does not have access to any strictly previously seen data. Instead, for a given natural number n, the learner may *query*, in parallel, for up to n different datapoints, whether those datapoints have

[1] For such a learner with a number of memory states equal $c \geq 1$, intuitively, the learner can store any one out of c different *values* in its long term memory [FKS95, KS95]. For example, when $c = 2^k$, the memory is equivalent to the learner having k bits of memory.

For the criteria studied for example in [Wie76, CJLZ99, JK09, JLMZ10], learning functions also have access to their just prior output conjecture (if any), *but*, for the criteria studied herein, learning functions have no such access.

been seen previously. No query may depend on the outcome of another query, and all queries will be individually answered truthfully, so that the learner knows for each queried datum whether it has been seen before.

In [CCJS07b], the authors show that, for each choice of parameter $n > 0$, U-shapes are necessary for the full learning power of MLF learning. As an open problem (Problem 39), they ask, for any given parameter $m > 0$, whether there is a parameter $n > m$ such that MLF learning with a (possibly high) parameter of n allows for non-U-shaped learning of all classes of languages that are MLF learnable with parameter m. We answer this question negatively, and show a much stronger result: Theorem 5 below says that there is a class of languages learnable memorylessly with a *single* feedback query which is not non-U-shapedly MLF learnable with *arbitrarily many sequential recall queries* For this, the learner may even continue asking queries, *dependent on the outcome of previous queries*, and not be limited to any finite number.

We complement this latter result by showing that any class of *infinite* languages learnable memorylessly with finitely many feedback queries is so learnable without U-shapes. Even stronger, Theorem 6 states that each TxtEx-learnable class of infinite languages is learnable memorylessly with arbitrarily many feedback queries and without U-shapes.

For this theorem, it is essential that the number of feedback queries is not bounded: Theorem 7 states that there is a class of infinite languages learnable memorylessly with a single feedback query, which is *not* learnable with*out* U-shapes by any *particular bounded* number of feedback queries.

We conclude our analysis of MLF learning by showing that it is *essential* that a query can be used to find out whether the current datum has been seen before (see Theorem 9).

Many proofs involve subtle infinitary program self-reference arguments employing the Operator Recursion Theorem (ORT) and variants from [Cas74, Cas94]. Because of space limitations, some proofs are omitted herein.

2 Mathematical Preliminaries

Unintroduced computability-theoretic notions follow [Rog67].

$\mathbb{N}$ denotes the set of natural numbers, $\{0, 1, 2, \ldots\}$.

The symbols $\subseteq, \subset, \supseteq, \supset$ respectively denote the subset, proper subset, superset and proper superset relation between sets. The symbol $\setminus$ denotes set-difference.

The quantifier $\forall^\infty x$ means "for all but finitely many x", the quantifier $\exists^\infty x$ means "for infinitely many x". For any set A, $\mathrm{card}(A)$ denotes its cardinality.

With $\mathfrak{P}$ and $\mathfrak{R}$ we denote, respectively, the set of all partial and of all total functions $\mathbb{N} \to \mathbb{N}$. With dom and range we denote, respectively, domain and range of a given function.

We sometimes denote a partial function f of $n > 0$ arguments $x_1, \ldots, x_n$ in lambda notation (as in Lisp) as $\lambda x_1, \ldots, x_n . f(x_1, \ldots, x_n)$. For example, with $c \in \mathbb{N}$, $\lambda x . c$ is the constantly c function of one argument.

For any predicate P, we let $\mu x . P(x)$ denote the least x such that $P(x)$.

We fix any computable 1-1 and onto pairing function $\langle \cdot, \cdot \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$.[2] Whenever we consider tuples of natural numbers as input to $f \in \mathfrak{P}$, it is understood that the general coding function $\langle \cdot, \cdot \rangle$ is used to (left-associatively) code the tuples into a single natural number.

If $f \in \mathfrak{P}$ is not defined for some argument x, then we denote this fact by $f(x)\uparrow$, and we say that f on x *diverges*; the opposite is denoted by $f(x)\downarrow$, and we say that f on x *converges*. If f on x converges to p, then we denote this fact by $f(x)\downarrow = p$.

We say that $f \in \mathfrak{P}$ *converges to* p iff $\forall^\infty x : f(x)\downarrow = p$; we write $f \to p$ to denote this.[3]

A partial function $\psi \in \mathfrak{P}$ is *partial computable* iff there is a deterministic, multi-tape Turing machine which, on input x, returns $\psi(x)$ if $\psi(x)\downarrow$, and loops infinitely if $\psi(x)\uparrow$. $\mathcal{P}$ and $\mathcal{R}$ denote, respectively, the set of all partial computable and the set of all computable functions $\mathbb{N} \to \mathbb{N}$. The functions in $\mathcal{R}$ are called *computable functions*.

We let φ be any fixed acceptable programming system for the partial computable functions $\mathbb{N} \to \mathbb{N}$ with associated complexity measure Φ. Further, we let φ_p denote the partial computable function computed by the φ-program with code number p, and we let Φ_p denote the partial computable *complexity* function of the φ-program with code number p.

A set $L \subseteq \mathbb{N}$ is *computably enumerable (*ce*)* iff it is the domain of a computable function. Let $\mathcal{E}$ denote the set of all ce sets. We let W be the mapping such that $\forall e : W(e) = \mathrm{dom}(\varphi_e)$. For each e, we write W_e instead of $W(e)$. W is, then, a mapping from $\mathbb{N}$ *onto* $\mathcal{E}$. We say that e is an index, or program, (in W) for W_e.

Whenever we consider sequences or finite sets as input to functions, we assume these objects to be appropriately coded as natural numbers.

2.1 Learning in the Limit

In this section we formally define several criteria for learning in the limit.

A *learner* is a partial computable function.

A *language* is a ce set $L \subseteq \mathbb{N}$. Any total function $T : \mathbb{N} \to \mathbb{N} \cup \{\#\}$ is called a *text*. For any given language L, a *text for* L is a text T such that $\mathrm{content}(T) = L$.

A *sequence generating operator* is an operator β taking as arguments a learner h and a text T and that outputs a function p. We call p the *learning sequence* of h given T.

Let β be a sequence generating operator and h a learner. We proceed by giving definitions for β-learning, and, additionally, for non-U-shaped variants. The non-U-shaped variants will require a learner never to change *semantically* any correct conjecture (on a path to successful learning).

We say that h *β-learns a language* L iff, for all texts T for L and $p = \beta(h, T)$, there is i_0 such that, for all $i \geq i_0$, $p(i) \in \{?, p(i_0)\}$ and $W_{p(i_0)} = L$. We denote the class of all languages β-learned by h with $\beta(h)$. By β we denote the set of all classes of languages β-learnable by learners.

[2] For a linear-time example, see [RC94, Section 2.3].

[3] $f(x)$ converges should not be confused with f converges *to*.

As the first example sequence generating operator we define **TxtEx** thus. $\forall h, T, i : \mathbf{TxtEx}(h, T)(i) = h(T[i])$. Then, for example, we see that h **TxtEx**-learns L iff, for all texts T for L, for some i_0, the sequence of learner h's outputs, $h(T[i_{i_0}]), h(T[i_{i_0}+1]), h(T[i_{i_0}+2]), \ldots$, begins with some correct W-program $p(i_0)$ for L and, after that the elements of the sequence are either $p(i_0)$ or ?.

We say that h **NU**β*-learns a language* L iff h β-learns L and, for all texts T for L and $p = \beta(h, T)$, for all i_0 such that $W_{p(i_0)} = L$ and all $i \geq i_0$, $W_{p(i)} = L$. We denote the class of all languages **NU**β-learned by h with **NU**$\beta(h)$. With **NU**β we denote the set of all classes of languages **NU**β-learnable by learners.

For Bounded Memory States learning with $n \geq 1$ states, learners are functions of the kind $\langle h, f \rangle$, i.e., learners with two outputs: the first for a new conjecture, the second for a new memory state. Given such a learner $\langle h, f \rangle$ and a text T, we define recursively the $\mathbf{BMS}_n$ learning sequence p and the sequence q of states[4] of $\langle h, f \rangle$ given T thus.

$$p(0) = ?; \tag{1}$$

$$q(0) = 0; \tag{2}$$

$$\forall i : p(i+1) = h(T(i), q(i)); \tag{3}$$

$$\forall i : q(i+1) = \min(n-1, f(T(i), q(i))). \tag{4}$$

The sequence generating operator $\mathbf{BMS}_n$ is defined accordingly.

Memoryless Feedback learning, as given in [CCJS07b], is a learning criterion where the learner works in two stages. In the first stage, the learner is presented a datum and uses it to compute a finite *set*. In the second stage, the learner computes a new conjecture, given the same datum and, additionally, for each element x of the finite set computed in the first stage, an indicator of whether x occurred previously in the current text.

Intuitively, each element x in the set resulting from the first stage represents the question "have I seen datum x previously?".

Variants of memoryless feedback learning, where the size of each such set is bounded by a fixed parameter $n \in \mathbb{N}$, are also studied in [CCJS07b]. Herein, we additionally study a variant where a learner is allowed to make queries *sequentially*, which we call memoryless *recall* learning (MLR).

We will not give formal details for modeling the associated sequence generating operators. Instead, we employ an informal query function `rcl` described below. For each $n \in \mathbb{N} \cup \{*\}$, let $\mathbf{MLF}_n$ be the sequence generating operator associated with allowing n *parallel* queries (feedback learning). Further, for all $m \in \mathbb{N} \cup \{*\}$, we let $\mathbf{MLR}_m$ be the sequence generating operator associated with allowing m *sequential* queries (recall learning).

As indicated, for specifying learners with feedback or recall queries, we introduce the use of `rcl` as follows.[5]

[4] Without loss of generality, the set of states is $\{0, \ldots, n-1\}$.

[5] The use of `rcl` is "syntactic sugar".

For $a, b, c \in \mathcal{P}$ and $d \in \mathbb{N}$ we will frequently make statements such as the following.

$$\forall x : \varphi_d(x) = \begin{cases} a(x), & \text{if } \mathtt{rcl}(c(x)); \\ b(x), & \text{otherwise.} \end{cases} \tag{5}$$

Intuitively, this means that φ_d on input (new datum) x will first recall $c(x)$, and then, if $c(x)$ was seen previously, output $a(x)$, otherwise $b(x)$. Furthermore, for a finite set D, we use $\mathtt{rcl}(D)$ to denote the set $\{x \in D \mid \mathtt{rcl}(x)\}$.

Starred Learners
For a learner h, possibly learning with restricted access to past data, we write $h^*(\sigma)$ for what the current output of h is after being fed the sequence σ of data items.

3 Bounded Memory States Learning (BMS)

Definition 1. Let $f \in \mathcal{P}$. For this definition, we let $f^* \in \mathcal{P}$ be such that $f^*(\emptyset) = 0$ and $\forall \sigma, x : f^*(\sigma \diamond x) = f(x, f^*(\sigma))$.[6]

For all $g \in \mathcal{R}$, let Y_g be such that

$$Y_g = \{j \mid (\forall k \leq j+1 : f^*(g[k])\downarrow) \wedge (\forall k \leq j) f^*(g[k]) \neq f^*(g[j+1])\}. \tag{6}$$

Intuitively, Y_g is the set of all j such that f, when presented the text g, changes into a previously not visited state after seeing element $g(j)$.

Lemma 2. Let $f \in \mathcal{P}$. Let f^* be as in Definition 1 above. For $g \in \mathcal{P}$, we will below refer to the following statement.

$$\forall k : f^*(g[k])\downarrow. \tag{7}$$

(i) There is an effective operator $\Theta : \mathcal{P} \to \mathcal{P}$ [7] such that, for all $g \in \mathcal{R}$, if (7), then $\Theta(g)$ is total and decides Y_g.
(ii) If range(f) is finite, then, for all $g \in \mathcal{R}$, Y_g is finite.
(iii) For all $g \in \mathcal{R}$, if (7), then

$$\forall \tau \subset g \; \exists \sigma \in \mathbb{S}\mathrm{eq}(\{g(j) \mid j \in Y_g\}) : f^*(\tau) = f^*(\sigma). \tag{8}$$

Proof. Obviously, Θ as follows satisfies (i).

$$\forall x, j : \Theta(\varphi_x)(j) = \begin{cases} \uparrow, & \text{if } (\exists k \leq j+1 : f^*(\varphi_x[k])\uparrow); \\ 1, & \text{else if } (\forall k \leq j) f^*(\varphi_x[k]) \neq f^*(\varphi_x[j+1]); \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

(ii) is easy to see.

[6] Note that $f^*(\emptyset) = 0$ is the initial state of any given **BMS**-learner.
[7] I.e. there exists a computable $f \in \mathcal{R}$ such that, for all φ-programs q, $\Theta(\varphi_q) = \varphi_{f(q)}$ [Rog67].

(iii) can be seen by $\subseteq$-induction on τ as follows. Let $\tau \subset g$ be such that, for all $\hat{\tau} \subset \tau$, $\exists \sigma \in \mathbb{S}\mathrm{eq}(\{g(j) \mid j \in Y_g\}) : f^*(\hat{\tau}) = f^*(\sigma)$. Let $\tau_0 \subseteq \tau$ be the $\subseteq$-minimum such that $f^*(\tau_0) = f^*(\tau)$. The conclusion is trivial if $\tau_0 = \emptyset$. Else, $\#\mathrm{elets}(\tau_0) - 1 \in Y_s$. Let $\sigma \in \mathbb{S}\mathrm{eq}(\{s(j) \mid j \in Y_s\})$ such that $f^*(\tau_0^-) = f^*(\sigma)$. Therefore,

$$f^*(\tau) = f^*(\tau_0) = f(\mathrm{last}(\tau_0), f^*(\tau_0^-)) = f(\mathrm{last}(\tau_0), f^*(\sigma)) = f^*(\sigma \diamond \mathrm{last}(\tau_0)). \tag{10}$$

Hence, $(\sigma \diamond \mathrm{last}(\tau_0)) \in \mathbb{S}\mathrm{eq}(\{g(j) \mid j \in Y_g\})$ is the desired sequence witnessing (iii) for τ. □

Contrasting $\mathbf{BMS}_1 = \mathbf{NUBMS}_1$ and $\mathbf{BMS}_2 = \mathbf{NUBMS}_2$ from [CCJS07b], we have the following theorem, solving an open problem, Problem 40, from [CCJS07b].

Theorem 3. We have

$$\mathbf{BMS}_3 \setminus \bigcup_{n>0} \mathbf{NUBMS}_n \neq \emptyset.$$

Proof. Let $M \in \mathcal{P}$ be such that

$$\forall v, x : M(x, v) = \begin{cases} \langle ?, v \rangle, & \text{if } x = \#; \\ \varphi_x(v), & \text{otherwise.} \end{cases} \tag{11}$$

Let $\mathcal{L} = \mathbf{BMS}_3(M)$. Let $n > 0$. Suppose, by way of contradiction, $\mathcal{L} \in \mathbf{NUBMS}_n$, as witnessed by $\langle h, f \rangle$ (h returns the new conjecture, f the new state). Suppose, without loss of generality, $\mathrm{range}(f)$ is finite. Let f^* be as in Definition 1 above. Let h^* be such that $h^*(\emptyset) = ?$ and $\forall \sigma, x : h^*(\sigma \diamond x) = h(x, f^*(\sigma))$. Let, for all $s \in \mathcal{R}$, Y_s be as in Definition 1 above, and let Θ be as shown existent in Lemma 2 (i).

Intuitively, $h^*(\sigma)$ is the hypothesis of the learner after seeing σ as input.

We define $\mathtt{ce}$ sets P, Q in uniform dependence of $r, s \in \mathcal{P}$ (we abbreviate, for all i, $s_i = \lambda j . s(i, j)$) such that $\forall a, b, \sigma, \tau$:

$$Q(b, \tau) \Leftrightarrow \exists \xi \in \mathbb{S}\mathrm{eq}(\mathrm{range}(\tau)) : \emptyset \neq W_{h^*(\tau \diamond r(b) \diamond \xi)} \cap (\mathrm{range}(s_b) \setminus \mathrm{range}(\tau)); \tag{12}$$

$$P(a, b, \sigma, \tau) \Leftrightarrow \begin{cases} a \neq b & \wedge \\ \sigma \in \mathbb{S}\mathrm{eq}(\{s_a(j) \mid \Theta(s_a)(j) = 1\}) & \wedge \\ \tau \in \mathbb{S}\mathrm{eq}(\mathrm{range}(s_b)) & \wedge \\ f^*(\sigma) = f^*(\tau) & \wedge \\ f^*(\sigma \diamond r(a)) = f^*(\tau \diamond r(b)) & \wedge \\ Q(b, \tau). \end{cases} \tag{13}$$

Fix a $\mathtt{ce}$-index for P. By 1-1 ORT, there are 1-1 $e, r, s, t, y, z \in \mathcal{R}$ with pairwise disjoint ranges and $p \in \mathbb{N}$ such that a number of restrictions are satisfied. The

first group of restrictions is given by the following four equations. $\forall x, i$:

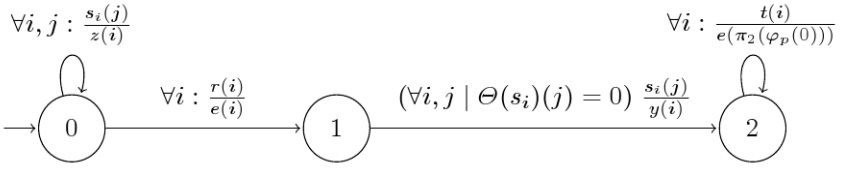
$$\varphi_p(x) = \mu\langle a, b, \sigma, \tau, d\rangle\,.\,[P(a, b, \sigma, \tau) \text{ in } \leq d \text{ steps}]; \tag{14}$$

$$W_{y(i)} = \text{range}(s_i) \cup \{r(i)\}; \tag{15}$$

$$W_{z(i)} = \text{range}(s_i); \tag{16}$$

$$W_{e(i)} = \{r(i)\} \cup \text{range}(t) \cup \begin{cases} \emptyset, & \text{if } \varphi_p(0)\uparrow; \\ \text{content}(\pi_3(\varphi_p(0))), & \text{else if } i = \pi_1(\varphi_p(0)); \\ \text{content}(\pi_4(\varphi_p(0))), & \text{otherwise.} \end{cases} \tag{17}$$

The second group of restrictions in our application of ORT is indicated by a labeled graph using vertices $\{0, 1, 2\}$. For all elements $x \in \text{range}(r) \cup \text{range}(s) \cup \text{range}(t)$ and $\ell \in \mathbb{N}$, an edge from vertex v to vertex w labeled $\frac{x}{\ell}$ (we use this kind of label for readability; $\frac{k}{\ell}$ is not to be confused with a fraction) in the graph just below adds the restriction $\varphi_x(v) = \langle \ell, w\rangle$ as part of our application of ORT.



The third and last group of restrictions is as follows.

$$(\forall i, j \mid \Theta(s_i)(j)\uparrow)\varphi_{s_i(j)}(1)\uparrow \tag{18}$$

and, for all $x \in \text{range}(r) \cup \text{range}(s) \cup \text{range}(t)$ and vertices v such that $\varphi_x(v)$ was not previously specified, we have the restriction $\varphi_x(v) = \langle ?, v\rangle$.

It is easy to verify that these three groups are not contradictory and embody a valid application of ORT.

The above graph now allows us to easily determine whether certain interesting subsets of $\text{range}(r) \cup \text{range}(s) \cup \text{range}(t)$ are in $\mathcal{L}$. For example,

$$\forall i \in \mathbb{N} : \text{range}(s_i) = W_{z(i)} \in \mathcal{L}. \tag{19}$$

Statement (19) can be derived from with the help of the graph, as the graph shows that, for all i, M, on any element from $\text{range}(s_i)$, stays in state 0 and outputs $z(i)$ as hypothesis.

From (19) we get, for all i and $\sigma \in \mathbb{S}\text{eq}(\text{range}(s_i))$, $f^*(\sigma)\downarrow$; in particular, for all i, we have (7) with s_i in the place of g. Therefore, for all i, using Lemma 2 (i), $\Theta(s_i)$ is total and (18) is vacuous. By Lemma 2 (ii), for each i, Y_{s_i} is finite, and, thus, the above graph easily shows

$$\forall i \in \mathbb{N} : \text{range}(s_i) \cup \{r(i)\} = W_{y(i)} \in \mathcal{L}. \tag{20}$$

Hence,

$$\forall i \in \mathbb{N}, \forall \rho \in \mathbb{S}\text{eq}(W_{y(i)}) : \langle h^*, f^*\rangle(\rho)\downarrow. \tag{21}$$

Claim 1. $\forall b \in \mathbb{N}\, \exists \tau \subset s_b : Q(b,\tau)$.

Proof of Claim 1. Let $b \in \mathbb{N}$. By the Pigeonhole Principle, as range(f) is finite, there is v such that $\exists^\infty k : f^*(s_b[k]) = v$. By (19), there is j_0 such that

$$W_{h^*(s_b[j_0])} = W_{z(b)}; \text{ and } \forall j \geq j_0 : h^*(s_b[j]) \in \{?, h^*(s_b[j_0])\}. \tag{22}$$

Let $j_1 > j_0$ be such that $f^*(s_b[j_1]) = v$. By (20), there is $k \in \mathbb{N}$ such that

$$W_{h^*(s_b[j_1] \diamond r(b) \diamond s_b[k])} = W_{y(b)}. \tag{23}$$

Let $j_2 > k$ be such that $f^*(s_b[j_2]) = v$. Then

$$W_{h^*(s_b[j_2] \diamond r(b) \diamond s_b[k])} = W_{y(b)}. \tag{24}$$

Hence, $\exists \tau \subset s_b : Q(b,\tau)$ as witnessed by $s_b[j_2]$ for τ and $s_b[k]$ for ξ.
□ (for Claim 1)

Claim 2. $\varphi_p(0)\downarrow$.

Proof of Claim 2. For the proof of this claim only, for each $b \in \mathbb{N}$, we fix τ_b as shown existent by Claim 1.

There are only finitely many pairs of states (elements of range(f)), while there are infinitely many $b \in \mathbb{N}$. Hence, by the Pigeonhole Principle, there are $a, b \in \mathbb{N}$ such that $a \neq b$, $f^*(\tau_a) = f^*(\tau_b)$ and $f^*(\tau_a \diamond r(a)) = f^*(\tau_b \diamond r(b))$. To show the claim, we use Lemma 2 (iii) with s_a in place of g to replace τ_a by σ such that $\sigma \in \mathbb{S}\mathrm{eq}(\{s_a(j) \mid j \in Y_{s_a}\})$ and $f^*(\tau_a) = f^*(\sigma)$. Thus,

$$f^*(\sigma \diamond r(a)) = f^*(\tau_a \diamond r(a)) = f^*(\tau_b \diamond r(b)) \tag{25}$$

and

$$f^*(\sigma) = f^*(\tau_a) = f^*(\tau_b). \tag{26}$$

Now we see that $P(a,b,\sigma,\tau_b)$, as

$$\begin{aligned}
a \neq b &\quad \text{by choice of } a, b;\\
\sigma \in \mathbb{S}\mathrm{eq}(\{s_a(j) \mid \Theta(s_a)(j) = 1\}) &\quad \text{by choice of } \sigma \text{ and } \Theta(s_a) \text{ decides } Y_{s_a};\\
\tau_b \in \mathbb{S}\mathrm{eq}(\mathrm{range}(s_b)) &\quad \text{as } \tau_b \subset s_b;\\
f^*(\sigma) = f^*(\tau_b) &\quad \text{as (26)};\\
f^*(\sigma \diamond r(a)) = f^*(\tau_b \diamond r(b)) &\quad \text{because of (25)};\\
Q(b,\tau) &\quad \text{by choice of } \tau_b.
\end{aligned}$$

□ (for Claim 2)

Let

$$\langle a, b, \sigma, \tau, d\rangle = \varphi_p(0), \tag{27}$$

and let ξ be as stated existent by $Q(b,\tau)$. We have

$$W_{e(a)} = \{r(a)\} \cup \mathrm{range}(t) \cup \mathrm{content}(\sigma), \text{ and} \tag{28}$$

$$W_{e(b)} = \{r(b)\} \cup \mathrm{range}(t) \cup \mathrm{content}(\tau). \tag{29}$$

It is easy to see (from the graph above) that $W_{e(a)}, W_{e(b)} \in \mathcal{L}$.

Let

$$\rho_a = \sigma \diamond r(a), \tag{30}$$
$$\rho_b = \tau \diamond r(b), \tag{31}$$
$$T_a = \rho_a \diamond t, \tag{32}$$
$$T_b = \rho_b \diamond t, \text{ and} \tag{33}$$
$$T'_b = \rho_b \diamond \xi \diamond t. \tag{34}$$

Then T_a is a text for $W_{e(a)}$ and T_b, T'_b are texts for $W_{e(b)}$. As $f^*(\rho_a) = f^*(\rho_b)$, and, as h has to identify $W_{e(a)}$ from $\rho_a \diamond t = T_a$ and $W_{e(b)}$ from $\rho_b \diamond t = T_b$, we have, for all k, $h^*(\rho_a \diamond t[k]) =?$ and $h^*(\rho_b \diamond t[k]) =?$. Thus, there is ℓ such that $W_{h^*(\rho_b[\ell])} = W_{e(b)}$.

To see that we have a U-shape with text T'_b:

1. $\exists \ell : W_{h^*(\rho_b[\ell])} = W_{e(b)}$ (as stated just above);
2. $W_{h^*(\rho_b \diamond \xi)} \neq W_{e(b)}$ (by ξ witnessing $Q(b, \tau_b)$); and
3. $\exists \ell : W_{h^*(\rho_b \diamond \xi \diamond t[\ell])} = W_{e(b)}$ (as T'_b is a text for $W_{e(b)} \in \mathcal{L}$).

This is a contradiction to $\mathcal{L} \in \mathbf{NUBMS}_n(\langle h, f \rangle)$. □

4 Memoryless Feedback Learning (MLF)

The main theorem in this section is Theorem 5 just below. This theorem answers the open question mentioned in Section 1.3 above regarding memory-less feedback learning.

Theorem 6 shows that memoryless learning with arbitrarily many feedback queries is equivalent to **TxtEx**-learning *on classes of infinite languages*.

Memoryless feedback learning, as defined above, trivially allows a learner to query for whether the *current input element* has been seen *strictly previously*. In Definition 8 below we give a variant of memoryless feedback learning, called MLF′ learning, where all queries are answered based on *all* data seen so far, *including* the current datum. For MLF′ learning, it is no longer possible to query to see if the current datum has been seen previously. From Theorem 9 we have that the learning power of MLF′ learning is strictly lower.

Some results in this section make use of the following definition.

Definition 4. Let $h_0 \in \mathcal{P}$ be such that $\forall i, x, D$:

$$h_0(i, x, D) = \begin{cases} \emptyset, & \text{if } x = \# \text{ and } i = 0; \\ ?, & \text{if } x = \# \text{ and } i \neq 0; \\ \varphi_x(i, D), & \text{otherwise.} \end{cases} \tag{35}$$

Let $\mathcal{L}_0 = \mathbf{MLF}_1(h_0)$.

Intuitively, h_0 is a learner which learns languages by interpreting each input datum as a program for the computations to make to get an appropriate output.

In fact, one can argue that not much information about how to identify $\mathcal{L}_0$ is actually in h_0; instead, one could say that "$\mathcal{L}_0$ learns itself." We call such classes of languages "self-learning classes of languages." Using such classes to show something is learnable with respect to one criterion and *not* with respect to some other is beneficial in two ways. We explain using $\mathcal{L}_0$ as an example. First, trivially, $\mathcal{L}_0$ is $\mathbf{MLF}_1$-learnable – $\mathcal{L}_0$ was defined to be the set of all languages so learnable by the explicitly given learner h_0. Secondly, one merely has to use recursion theorems, such as variants of ORT, to diagonalize out of $\mathcal{L}_0$. We so use the self-learning $\mathcal{L}_0$ in the proofs of Theorems 5 and 9 (but the latter proof is omitted herein) and we employ a variant in the (omitted) proof of Theorem 7.

Theorem 5

$$\mathbf{MLF}_1 \setminus \mathbf{NUMLR}_* \neq \emptyset.$$

Proof. We consider $\mathcal{L}_0$ from Definition 4. Suppose $h_1 \in \mathcal{P}$ $\mathbf{MLR}_*$-learns $\mathcal{L}_0$. We show that h_1 is *not* non-U-shaped. It is easy to see that we can assume, without loss of generality,

$$\forall \tau, x : h_1^*(\tau \diamond x \diamond x) \in \mathbb{N} \Rightarrow h_1^*(\tau \diamond x) = h_1^*(\tau \diamond x \diamond x). \tag{36}$$

Let $f \in \mathcal{P}$ be such that, on input x, f first computes $h_1(x)$ where all queries are answered with "false". If $h_1(x)\downarrow$, f outputs the maximum recalled element plus 1 (or 0, if no queries were asked). Then we have

$$\forall x : h_1^*(x)\downarrow \Rightarrow f(x)\downarrow \text{ and } h_1^* \text{ on } x \text{ does not recall any } y \geq f(x). \tag{37}$$

By padded ORT, there are $e_0, e_1, a \in \mathbb{N}$ and strictly monotonic increasing functions $\hat{b}, \hat{c} \in \mathcal{R}$ such that $\hat{b}$ and $\hat{c}$ have disjoint ranges, neither containing a, and, abbreviating

$$\begin{aligned} b &= \hat{b}(f(a)); \\ c &= \lambda i . \hat{c}(i + f(a)); \\ E &= \{c(i) \mid \exists j \geq i : h_1^*(a \diamond c[j])\downarrow \neq h_1^*(a \diamond c[j+1])\downarrow\} \end{aligned}$$

we have $\forall i, x$:

$$W_{e_0} = \{a\} \cup \text{range}(c); \tag{38}$$

$$W_{e_1} = \begin{cases} \{a\}, & \text{if } h_1^*(a)\uparrow \text{ or } h_1^*(a) =?; \\ \{a, b\} \cup E, & \text{otherwise;} \end{cases} \tag{39}$$

$$\varphi_a(x) = \begin{cases} ?, & \text{if } \texttt{rcl}(a); \\ e_1, & \text{otherwise;} \end{cases} \tag{40}$$

$$\varphi_{\hat{b}(i)}(x) = \begin{cases} ?, & \text{if } \texttt{rcl}(\hat{b}(i)); \\ e_1, & \text{otherwise;} \end{cases} \tag{41}$$

$$\varphi_{\hat{c}(i)}(x) = \begin{cases} e_1, & \text{if } \texttt{rcl}(b); \\ e_0, & \text{otherwise.} \end{cases} \tag{42}$$

Claim 3. $h_1^*(a) \in \mathbb{N}$.

Proof of Claim 3. Suppose, by way of contradiction, otherwise. We have $\{a\} = W_{e_1} \in \mathcal{L}_0$. If $h_1^*(a)\uparrow$, then h_1 would not learn $W_{e_1} \in \mathcal{L}_0$, a contradiction. Hence, $h_1^*(a) =?$. Using (36), we get that h_1, on the text $\lambda i.a$, does not learn $\{a\} = W_{e_1} \in \mathcal{L}_0$, a contradiction. □ (FOR CLAIM 3)

Using (37) and Claim 1, we have $f(a)\downarrow$. Hence, b is defined and c is total. It is now easy to see that

$$W_{e_0} \in \mathcal{L}_0. \tag{43}$$

Therefore, h_1 $\mathbf{MLR}_*$-learns W_{e_0}. Let k be minimal such that

$$\forall i \geq k : h_1^*(a \diamond c[i]) \in \{?, h_1^*(a \diamond c[k])\}. \tag{44}$$

In particular,

$$W_{h_1^*(a \diamond c[k])} = W_{e_0}. \tag{45}$$

Hence, W_{e_1} is the finite set $\{a, b, c(0)\} \cup \text{content}(c[k])$. It is easily verified that $W_{e_1} \in \mathcal{L}_0$.

Note that, as $\hat{b}$ and $\hat{c}$ are strictly monotone increasing,

$$\forall x \in \{b\} \cup \text{range}(c) : x \geq f(a). \tag{46}$$

Hence, by choice of f (see (37)), h_1 on a cannot recall any $x \in \{b\} \cup \text{range}(c)$. Therefore,

$$h_1^*(a) = h_1^*(c[k] \diamond b \diamond a). \tag{47}$$

Claim 4. $W_{h_1^*(a)} = W_{e_1}$.

Proof of Claim 4. From Claim 1 we have that $h_1^*(a) \in \mathbb{N}$. As h_1 $\mathbf{MLR}_*$-identifies W_{e_1} from the text $c[k] \diamond b \diamond \lambda i.a$ and using (36),

$$W_{h_1^*(c[k] \diamond b \diamond a)} = W_{e_1}. \tag{48}$$

Using (47), we get the claim. □ (FOR CLAIM 4)

We consider the text $T = a \diamond c[k] \diamond b \diamond \lambda x.\#$ for W_{e_1}. The following shows that h_1 has a U-shape on T.

1. $W_{h_1^*(a)} = W_{e_1}$ by Claim 2;
2. $W_{h_1^*(a \diamond c[k])} = W_{e_0}$ by (45);
3. $\exists j \geq k+1 : W_{h_1^*(T[j])} = W_{e_1}$ as h_1 $\mathbf{MLR}_*$-identifies W_{e_1}.

Therefore, h_1 is not non-U-shaped on $\mathcal{L}_0$. □

With $\text{Pow}(\mathcal{E}_\infty)$ we denote the powerset of all infinite ce sets.

Theorem 6

$$\text{Pow}(\mathcal{E}_\infty) \cap \mathbf{NUMLF}_* = \text{Pow}(\mathcal{E}_\infty) \cap \mathbf{TxtEx}.$$

Theorem 7

$$(\mathbf{MLF}_1 \cap \mathrm{Pow}(\mathcal{E}_\infty)) \setminus \bigcup_{n \in \mathbb{N}} \mathbf{NUMLF}_n \neq \emptyset.$$

Note that **MLF** allows for a learner to determine (at the cost of a query) whether the current datum has been seen previously (i.e., is a repetition), and the previous proofs in this section sometimes made use of this ability. The next theorem states that this ability is important for learning power.

Definition 8. Call $\mathbf{MLF}'$ the sequence generating functional that one gets by modifying **MLF** to answer *true* to recalls for the current datum.

The sequence generating functional $\mathbf{MLF}'$ destroys a learners ability to determine whether the current datum has been seen previously (i.e., is a repetition).

Theorem 9

$$\mathbf{MLF}_1 \setminus \mathbf{MLF}'_* \neq \emptyset.$$

References

[BCM+08] Baliga, G., Case, J., Merkle, W., Stephan, F., Wiehagen, W.: When unlearning helps. Information and Computation 206, 694–709 (2008)

[Car82] Carey, S.: Face perception: Anomalies of development. In: Strauss, S., Stavy, R. (eds.) U-Shaped Behavioral Growth. Developmental Psychology Series. Academic Press, NY (1982)

[Cas74] Case, J.: Periodicity in generations of automata. Mathematical Systems Theory 8, 15–32 (1974)

[Cas94] Case, J.: Infinitary self-reference in learning theory. Journal of Experimental and Theoretical Artificial Intelligence 6, 3–16 (1994)

[Cas99] Case, J.: The power of vacillation in language learning. SIAM Journal on Computing 28, 1941–1969 (1999)

[CCJS07a] Carlucci, L., Case, J., Jain, S., Stephan, F.: Non-U-shaped vacillatory and team learning. Journal of Computer and System Sciences, Special Issue in Memory of Carl Smith (2007)

[CCJS07b] Carlucci, L., Case, J., Jain, S., Stephan, F.: Results on memory-limited U-shaped learning. Information and Computation 205, 1551–1573 (2007)

[CJLZ99] Case, J., Jain, S., Lange, S., Zeugmann, T.: Incremental concept learning for bounded data mining. Information and Computation 152, 74–110 (1999)

[CL82] Case, J., Lynes, C.: Machine inductive inference and language identification. In: Nielsen, M., Schmidt, E.M. (eds.) ICALP 1982. LNCS, vol. 140, pp. 107–115. Springer, Heidelberg (1982)

[FJO94] Fulk, M., Jain, S., Osherson, D.: Open problems in Systems That Learn. Journal of Computer and System Sciences 49, 589–604 (1994)

[FKS95] Freivalds, R., Kinber, E., Smith, C.: On the impact of forgetting on learning machines. Journal of the ACM 42, 1146–1168 (1995)

[Gol67] Gold, E.: Language identification in the limit. Information and Control 10, 447–474 (1967)

[JK09] Jain, S., Kinber, E.: Iterative learning from texts and counterexamples using additional information. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 308–322. Springer, Heidelberg (2009)

[JLMZ10] Jain, S., Lange, S., Moelius III, S.E., Zilles, S.: Incremental learning with temporary memory. Theoretical Computer Science 411, 2757–2772 (2010)

[JORS99] Jain, S., Osherson, D., Royer, J., Sharma, A.: Systems that Learn: An Introduction to Learning Theory, 2nd edn. MIT Press, Cambridge (1999)

[KS95] Kinber, E., Stephan, F.: Language learning from texts: Mind changes, limited memory and monotonicity. Information and Computation 123, 224–241 (1995)

[LZ96] Lange, S., Zeugmann, T.: Incremental learning from positive data. Journal of Computer and System Sciences 53, 88–103 (1996)

[MPU+92] Marcus, G., Pinker, S., Ullman, M., Hollander, M., Rosen, T.J., Xu, F.: Overregularization in Language Acquisition. In: Monographs of the Society for Research in Child Development, vol. 57(4). University of Chicago Press, Chicago (1992); Includes commentary by H. Clahsen

[OSW86] Osherson, D., Stob, M., Weinstein, S.: Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists. MIT Press, Cambridge (1986)

[OW82] Osherson, D., Weinstein, S.: Criteria of language learning. Information and Control 52, 123–138 (1982)

[RC94] Royer, J., Case, J.: Subrecursive Programming Systems: Complexity and Succinctness. In: Research monograph in Progress in Theoretical Computer Science. Birkhäuser, Basel (1994)

[Rog67] Rogers, H.: Theory of Recursive Functions and Effective Computability. McGraw Hill, New York (1967); Reprinted by MIT Press, Cambridge, Massachusetts (1987)

[SS82] Strauss, S., Stavy, R. (eds.): U-Shaped Behavioral Growth. Developmental Psychology Series. Academic Press, NY (1982)

[TA02] Taatgen, N.A., Anderson, J.R.: Why do children learn to say broke? A model of learning the past tense without feedback. Cognition 86, 123–155 (2002)

[WC80] Wexler, K., Culicover, P.: Formal Principles of Language Acquisition. MIT Press, Cambridge (1980)

[Wie76] Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. Elektronische Informationverarbeitung und Kybernetik 12, 93–99 (1976)