Information and Computation 220–221 (2012) 60–73

Contents lists available at SciVerse ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Learning secrets interactively. Dynamic modeling in inductive inference

John Case[a], Timo Kötzing[b,*]

[a] Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716, USA
[b] Department 1: Algorithms and Complexity, Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany

ARTICLE INFO

Article history:
Received 23 July 2012
Revised 1 October 2012
Available online 23 October 2012

Keywords:
Inductive inference
Interactive learning

ABSTRACT

Introduced is a new inductive inference paradigm, *dynamic modeling*. Within this learning paradigm, for *example*, function h *learns* function g iff, in the i-th iteration, h and g both produce output, h gets the sequence of all outputs from g in prior iterations as input, g gets all the outputs from h in prior iterations as input, and, from some iteration on, the sequence of h's outputs will be *programs for* the *output sequence* of g.

Dynamic modeling provides an idealization of, for example, a social interaction in which h seeks to discover program models of g's behavior it sees in interacting with g, and h *openly* discloses to g its sequence of candidate program models to see what g says back.

Sample results: every g can be so learned by some h; there are g that can only be learned by an h if g can also learn that h back; there are extremely secretive h which cannot be learned back by any g they learn, but which, nonetheless, succeed in learning infinitely many g; quadratic time learnability is strictly more powerful than linear time learnability. This latter result, as well as others, follows immediately from general correspondence theorems obtained from a *unified* approach to the paradigms within inductive inference.

Many proofs, some sophisticated, employ machine self-reference, a.k.a., recursion theorems.

1. Introduction and motivation

In **Computational limit learning of computable functions** mapping the non-negative integers to the same, an *algorithmic learner* is iteratively given more and more *finite* information generated by a *computable target function*. From this information, the learner, in each iteration (may) synthesize a (suitably interpreted) natural number as output. In the literature, many criteria of successful learning have been proposed. Each such learning criterion defines precisely, possibly among other things, in what way the information will be generated by the target function and how the sequence of iteratively generated outputs and the target function have to relate for the learning to be considered successful. Sometimes each output number will be interpreted as (numerically naming) a program, other times each number will represent a prediction for a yet unseen data point. It is helpful for the present paper to briefly consider some illustrative examples.

Ex-learning [17] exemplifies the category of *identification*. h Ex-learns target g iff, in the i-th iteration, h outputs a conjecture on input $g(0)\ldots g(i-1)$ and there are j, e such that $\forall k \geqslant j$: $h(g(0)\ldots g(k-1)) = e$ and e is a program for g.[1] Such a program e could be carried away and used *offline*.

Learning the next value (Nv-learning) [1,4] exemplifies the category of *extrapolation*. h Nv-learns target g iff, h is total and, in the i-th iteration, h outputs a conjecture on input $g(0)\ldots g(i-1)$ and there is a j such that $\forall k \geqslant j$: $h(g(0)\ldots g(k-1)) = g(k)$.[2] The successful extrapolants $h(g(0)\ldots g(k-1))$, $k \geqslant j$, can be used *online*.

* Corresponding author.
E-mail addresses: case@cis.udel.edu (J. Case), koetzing@mpi-inf.mpg.de (T. Kötzing).

[1] The term 'Ex' stands for *explanatory* [11].
[2] The term 'Nv' stands for *next value* [1].

0890-5401/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ic.2012.10.002

Learning to coordinate (Coord-learning) [22] exemplifies the category of *coordination*. h Coord-learns a target g iff, in the i-th iteration, h and g both produce output, h gets the sequence of all outputs from g in prior iterations as input, g gets all the outputs from h in prior iterations as input, and, from some iteration on, the sequence of h's outputs will be the same as the sequence of g's outputs. The finally successfully coordinated matching outputs can be used *online*.

In these examples we see that, while Coord-learning features a *reactive learnee* g, Ex- and Nv-learning feature a *passive learnee* g.

	Passive learnee	Reactive learnee
Learning programs	Identification	??
Predicting next value	Extrapolation	Coordination

In the just above table, there is a missing, not heretofore studied category entry for *offline* with *reactive learnee*. We refer to this category as *dynamic modeling*, and it is the subject of the present paper. **XBc**-learning exemplifies the category of *dynamic modeling*. h **XBc**-learns a target g iff, in the i-th iteration, h and g both produce output, h gets the sequence of all outputs from g in prior iterations as input, g gets all the outputs from h in prior iterations as input, and, from some iteration on, the sequence of h's outputs will be *programs for* the *output sequence* of g.[3]

In cognitive science, *theory of mind* refers to ones having a model (or models) of another's thoughts, emotions, and perspectives — including those different from ones own. Ideally, one might have a *program* (or programs) generating the behavior of the other, but — the behavior presented by the other would, in reality, be all and only that resulting from crossfeeding between oneself and the other. While one is attempting to synthesize program(s) for the other, *a* technique to employ is to pass on a sequence of remarks such as, "I think you are like ..." (where ... might be a program), and, then, attend to the resultantly elicited sequence of reactions of the other — as one formulates further programs/models of the other. Of course, in reality, one might, in seeking social understanding, carry out variants, including highly filtered variants, of the just above scenario. The unfiltered and very idealized scenario above is, nonetheless, covered by dynamic modeling.

Next we summarize the contents of the remaining sections.

In Section 2 below we present mathematical preliminaries.

Section 3 presents a *unified* approach to limiting learning criteria. This pays off in Section 5 where we can *then* provide general results applying to many criteria at once and, thereby, *quickly* obtain some nice corollaries.[4]

Section 4 involves cooperativeness vs. secretiveness in dynamic modeling. Considered are dynamic modelers which may or may not, in return, be dynamically modeled themselves. Proposition 4.1 implies that *no* computable g can keep models of its behavior totally secret; moreover, for any computable g, there are infinitely many constant functions h that **XBc**-learn g.[5]

Surprisingly, Theorem 4.3 implies that there is a computable g so that, *no* computable h *that* **XBc**-*learns* g can keep models of its behavior a secret from g, i.e., such h gives itself away: g, in turn, **XBc**-learns h. Positively, such a g is, then, *extremely cooperative*: informally, g can figure out the behavior of any computable h that figures out its behavior. Furthermore, such a g can be chosen to be linear time computable! The proof of Theorem 4.3 is particularly elegant.

We say that computable h is *extremely uncooperative* iff there is *no* computable g such that h **XBc**-learns g and g **XBc**-learns h. Theorem 4.7 implies there are extremely uncooperative computable h which, nonetheless, are infinitely successful, i.e., such that h **XBc**-learns infinitely many computable g.

Results in Section 4 feature open disclosure of certain learners' models of another while not disclosing their own models to the other. For comparison and contrast, a *zero-knowledge proof* [5] permits open, convincing disclosure of its existence without disclosing how it works.

Section 5 features two general and powerful correspondence theorems (Theorems 5.10 and 5.12) regarding many of the criteria discussed above and in Section 3. A further result is *dynamic modeling by enumeration* as given in Theorem 5.3. This theorem states that each set of uniformly computable total functions is learnable in the sense of dynamic modeling; this is a known and easy result for Ex-learning, but it requires some thought in the setting of dynamic modeling.

Theorem 5.10 *immediately* yields Corollary 5.11 which implies, for *example*, that quadratic time **XBc**-learnability is strictly more powerful than lintime **XBc**-learnability.[6]

Theorem 5.12 *immediately* yields Corollary 5.13 which provides a number of learning criteria hierarchies and separations. An *example*: the powers of **XBc**-learning and of Coord-learning are incomparable.

Many proofs, some sophisticated, employ machine self-reference techniques, including Kleene Recursion Theorem (**KRT**) [25, page 214, problem 11-4] and Case's Operator Recursion Theorem (**ORT**) [6,7]. The latter achieves infinitary self (and other) reference.

This paper is an extension of the conference paper [10] and was part of the second author's PhD thesis [19].

[3] We use the cross-shaped **X** to denote cross-feeding. Of course crossfeeding of data is common to both the categories of coordination and dynamic modeling. In Section 3 we use the **X** also in talking about the former category. **Bc** stands for *behaviorally correct* [11].

[4] In Section 3 Ex-learning will be called **GEx**-learning, where the **G** is for Gold [17]. Nv-learning will be called $\mathcal{R}$**GM**, where the $\mathcal{R}$ is for (total) computable learner and learnee, and the **M** is for Matching. Coord-learning will be called **XM**-learning.

[5] Actually, Proposition 4.1 is stated for a more restrictive criterion within the dynamic modeling category.

[6] Nothing like this happens for, for example, Ex-learning, since by an extension of Pitt's postponement tricks from [23] (otherwise unrestricted) Ex-learning with lintime learners is just as powerful as Ex-learning.