Limits of Computational Learning

Timo Kötzing

Summer term 2012

April 18, 2012

General Information

- Lectures: Wednesdays, 10-12 in 024 (MPII building).
- Exercises: Fridays, 10-12 in 023 (MPII building),
- Exercises start April 27.
- Some classes will be cancelled (May 2 and July 11).
- Homework handed out every Wednesday, due following Wednesday.
- Grading is 50% homework and 50% final exam.
- Successful participation earns 6 credits.

Content of the Course

- We will start with two weeks of computability theory.
- Then we learn about function learning in the limit.
- At some point, we will discuss language learning in the limit.
- I will put my lecture notes online (subjecto to changes).
- Promise: All homework will be doable with online lecture notes alone.

What do you know about?

- Do you know an abstract machine model, like Turing machines?
- Do you know the halting problem?
- Do you know a recursion theorem?
- Do you know the parameter theorem (or s-m-n theorem)?

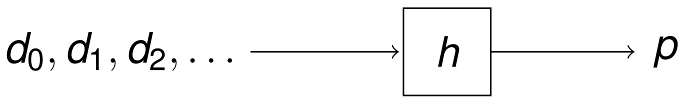
Function Learning

Learning programs for given sequences:

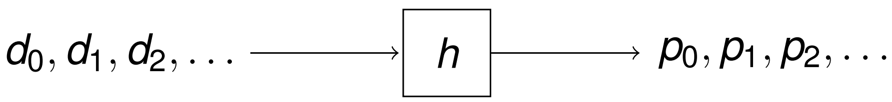
$0, 1, 2, 3, 4, 5, \ldots$	$x \mapsto x$
$0, 1, 4, 9, 16, 25, \ldots$	$x \mapsto x^2$
$0, 1, 0, 0, 0, 0, \ldots$	$x \mapsto \begin{cases} 1 & \text{if } x = 1 \\ 0 & \text{otherwise.} \end{cases}$
$0, 1, 1, 2, 3, 5, 8, 13, \ldots$	$x \mapsto \mathrm{fib}(x)$

Finite and Limit Learning

- One-Shot or Finite Learning:



- Multi-Shot, Trial-and-Error or Limit Learning:



Formal Definitions

- Let $\mathbb{N} = \{0, 1, 2, \ldots\}$.
- For $g : \mathbb{N} \to \mathbb{N}$ and $k \in \mathbb{N}$ let $g[k] = g(0), \ldots, g(k-1)$.
- If h is a learner and g is a learnee, we define the learning sequence p of h on g by

$$\forall k : p(k) := h(g[k]).$$

- We say that h **Ex**-learns g iff, for some i, $p(i)$ computes g and $p(i) = p(i+1) = p(i+2) = \ldots$.
- A set of computable functions is called **Ex**-learnable iff there is a learner learning every function in that set.
- For example: The set of all polynomial time computable functions is **Ex**-learnable.

Learning and Learning Time

- We sometimes want h to be computable in polytime (or some other time bound).
- Fact: For every **Ex**-learnable set of functions $\mathcal{S}$, there is such a polytime computable learner learning $\mathcal{S}$.
- Why? A polytime learner can be obtained from postponing necessary computations to a later time, where sufficient computing time is available (due to having a longer input).

Consistency

- h is called consistent iff $\forall g, k,\ h(g[k])$ correctly computes $g[k]$.
- Fact: There are **Ex**-learnable sets, not learnable by a consistent learner.
- Fact: The set of all polynomial time computable functions is **Ex**-learnable by a consistent polytime learner.
- Fact: The set of all exponential time computable functions is not **Ex**-learnable by a consistent polytime learner.

Iterative Learning

- h is called iterative iff it takes only one new datum plus its previous conjecture as input.
- Fact: There are **Ex**-learnable sets, not learnable by an iterative learner.
- Fact: Every consistently learnable set is iteratively learnable.