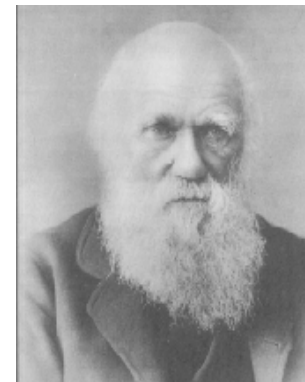
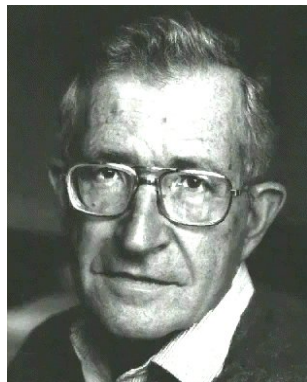


# Grammar acquisition – review and synthesis

Jelle Zuidema

Language Evolution & Computation, University of Edinburgh, UK  
(Funded by Prince Bernhard Culture Fund, Amsterdam, the Netherlands)



## Gold: identification in the limit

*“A class of language will be called identifiable in the limit [...] if there is an effective learner [...] with the following property: Given any language of the class and given any allowable training sequence for this language, the language will be identified in the limit.”*

- Positive evidence: *Text*
- Negative evidence: *Informant sequence*

## Context-free grammars are not learnable from text

Infinite languages can not be identified, because there exists an infinite sequence of finite languages that are indistinguishable for any amount of training samples.. I.e. no matter how many examples you have seen, you'll never know whether you've seen the whole language or whether you should generalize to (infinitely) more.

<i>infinite language</i>	<i>finite languages</i>
$S \mapsto Sa$	$S \mapsto a$
$S \mapsto a$	$S \mapsto aa$
	$S \mapsto aaa$
	$S \mapsto aaaa$
	$S \mapsto aaaaa$
	...

## Principle & Parameter grammars are learnable from text

E.g., by *identification through enumeration*. The algorithm considers a finite number of hypotheses; it sticks to an hypothesis until it receives a counter example; it will always receive a counter example within a finite amount of time. If it considers hypotheses in the right order, it will therefore always arrive and stay at the correct hypothesis within a finite amount of time.

$S \mapsto aSb$	$S \mapsto aS aA$
$S \mapsto ab$	$A \mapsto Ab b$
$aabb$	$aabb$
$aaaabbbb$	$aaaabbbb$
$aaaaaabbabbbb$	$aaaaaabbabbbb$
$aaaaabbbb$	$aabbbb$

## Learnability – overview

	Learnability criteria	Primary Data	Classes	
Gold67	IDL	Text	Super-finite	-
	IDL	Text + Informant	Super-finite	+
Horning69	Productivity	Text	stochastic cfg	+
H&W73	IDL	Text + Semantics	cfg	+
W&C80	Approx. correct	Text	cfg	-
	Approx. correct	Text	P&P	-
Val84	PAC	Text	cfg	-
Ang80	IDL	Text	fin. elasticity	+
Arb80	IDL	Phrase-structure	cfg	+

Hamburgh & Wexler, 1973; Wexler & Culicover, 1980; Valiant, 1984; Angluin, 1980, Arbib, 1980.

## What does it mean?

*“The basic results of the field [of learnability theory] include the formal, mathematical demonstration that without serious constraints on the nature of human grammar, no possible learning algorithm can in fact learn the class of human grammars.” (Wexler, 1999, “Innateness of Language”, MITECS)*

## What does it really mean?

*“It seems that some psychologists, suspicious of the innateness claims which have provided the intellectual backdrop to so much of the progress in modern linguistics, have found it difficult to give up on the belief that linguistic environments really do have properties (if only we could identify them) which would enable us to see them as providing a sufficient basis for grammar induction. We can be fairly confident in our conclusions under this heading, but we owe it to the skeptics to provide some justification for this confidence.” — Martin Atkinson, “Learnability and the acquisition of syntax” (Bertolo, 2001).*

## What does it really mean?

*“the variant of the argument from the poverty of stimulus that has emerged from the field of learnability theory is based on firm, formal results that show that learning is impossible without proper and interrelated assumptions on what grammars are possible in the first place, and what primary data is available.” (Pure empiricism is untenable)*

- Nothing in these formal results shows that the necessary constraints are language-specific adaptations; they could be simply generic properties of the human brain.
- Moreover, the nature of the available linguistic data is an empirical and not a formal issue.

## Conclusion

Learnability theory =  $\text{\LaTeX}$  fetishism

## Grammar induction

Stolcke, Adriaans, Kirby, Langley, Van Zaanen, Zuidema

*Frequent substrings are building blocks; building block that occur in the same context can be used interchangeably*

1. Incorporation: include an encountered string
2. Compression / Split: substitute frequent and long substrings with a non-terminal
3. Generalization / Merge: equate two nonterminals if they occur frequently in the same context

## Learnability revisited

If you have a model of:

- I-language and E-language (i.e. representation, production and interpretation),
- primary linguistic data
- and an acquisition procedure,

testing whether it leads to learnability is an important test for the validity of your model.

If that test is hard to pass, passing it is strong evidence. If it is easy to pass, passing it is still necessary, but might not tell you much about whether your theory is “right”.

## The learnability test

For every learning algorithm, one can in principle work out the classes of languages that it can learn. E.g. Adriaans: *shallow context-free languages*.

To guarantee success, you should present the learner only with languages that are in that class.

In iterated learning, the necessary constraints on the set of targets automatically emerge

- If learnability is *binary*, unlearnable languages will disappear after the first cycle;
- If learnability is *continuous*, languages might become gradually better learnable.

## Learnability $\wedge$ Expressiveness

Crucial question: can the learnability constraints and the expressiveness constraints be reconciled? I.e. do the language that emerge in iterated learning fulfill the requirements from your theory on language use?

## A new grammar induction algorithm

Some requirements

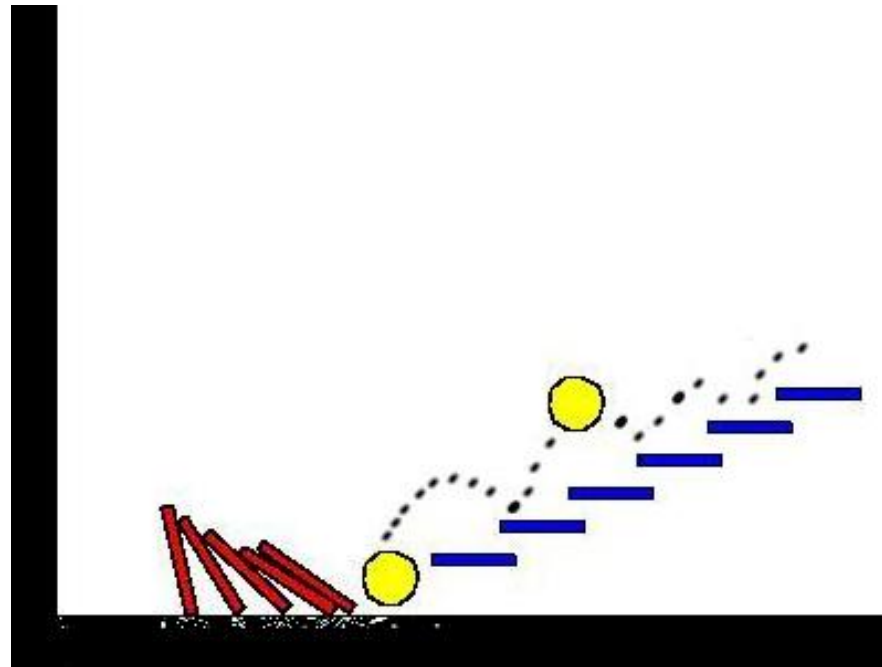
Expressiveness:

- Semantics of at least the expr. power of a high-order predicate logic
- Syntax of at least the expressive power of context-free grammars

Learnability:

- natural 'text', without POS tagging, phrase-structure, word-boundaries
- grounded & noisy meanings, derived from a shared context
- finite number of training samples
- communicative accuracy as the learnability criterion

## A shared world



## Semantics: predicate-logic / $\lambda$ -calculus

A meaning is a flat list of predicates, with a (possible empty) list of lamda's and a head. E.g.:

```
 $\lambda x \lambda y (x \mid (\text{approach } z) (\text{arg1 } z \ x) (\text{arg2 } z \ y))$ 
```

When applied to the following semantic description

```
(p  $\mid$  (circle p))
```

the resulting description is as follows:

```
 $\lambda y (p \mid (\text{approach } z) (\text{arg1 } z \ p) (\text{arg2 } z \ y)) (\text{circle } p)$ 
```

## Syntax: (pure) categorial grammars

A category is either

- of the basic categories  $n$  or  $s$
- of the structure (yields needs constraint),
  - yields and needs are categories
  - constraint is  $l$  (left),  $r$  (right) or  $a$  (anywhere)

## The induction algorithm

1. Incorporation: when receiving a training sample  $\langle f, m \rangle$ , add an association  $\langle f, m, S \rangle$  to the lexicon.
2. Split: align two lexical entries of the same type. If there is a shared valid substring  $t$  AND a shared valid set of predicates  $p$ , create four new lexical entries:
  - (a)  $\langle t, x | p, c' \rangle$
  - (b)  $\langle \hat{t}, \lambda(h | \hat{p}), (c \ c' \ L) \rangle$
  - (c)  $\langle \hat{t}, x | p, (c \ c'' \ L) \rangle$
  - (d)  $\langle t, \lambda(h | \hat{p}), c'' \rangle$

3. Merge

4. Decrease costs of an association if successfully used using Batali's rule.