# What Machines Can Never Learn: Models and Limitations

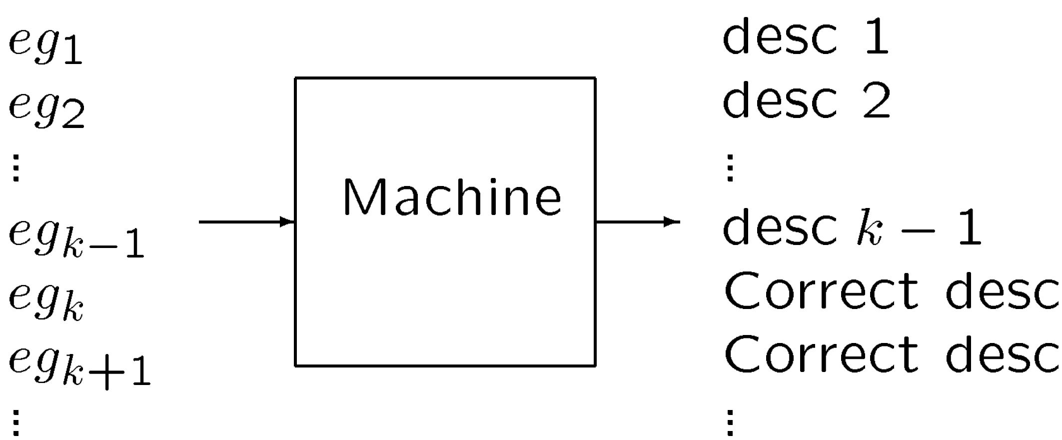
James A. Foster
Laboratory for Applied Logic
Department of Computer Science
University of Idaho

# Outline

- What is learning?
- Learning what?
- What is a machine?
- Various models of learning.

# What is learning?

## Learning From Examples




Given examples of target concept, Converge to correct description

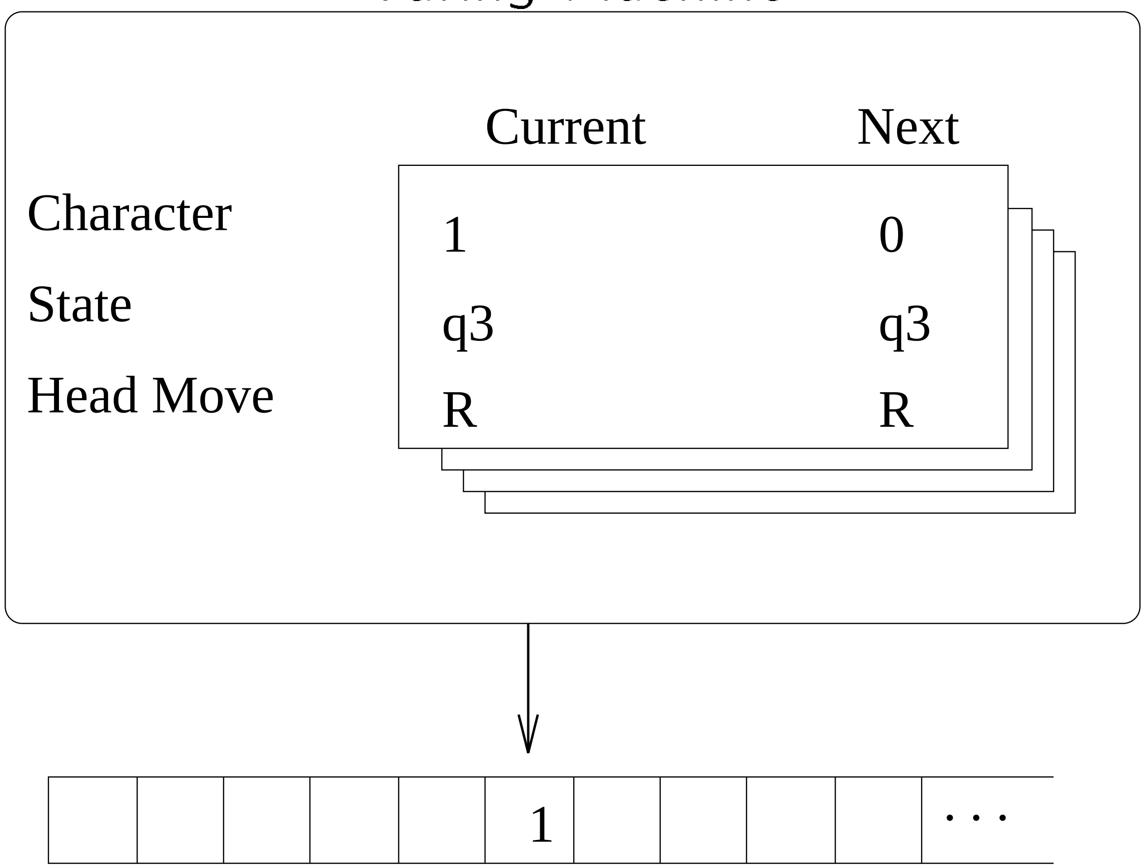
## Learning what?

## Functions on natural numbers

- Model *any* discrete domain
- Well-developed theory
- Very general

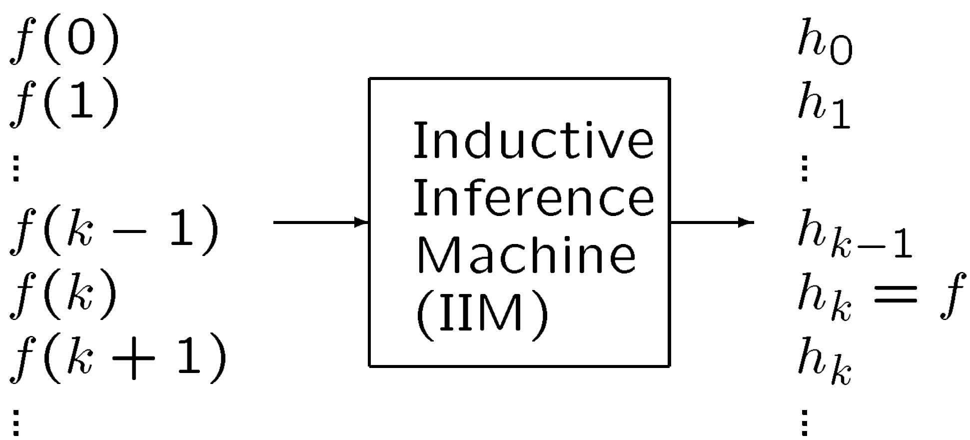
# What is a machine?

Turing Machine



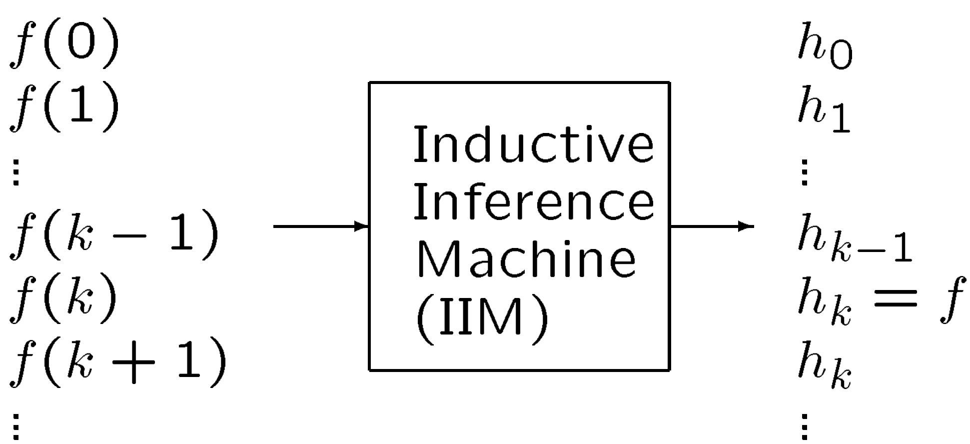

Or any reasonable programming language

# (Gold 67) Learning in the Limit




IIM eventually converges to index for $f$

# Syntactic (EXplanatory) Learning



Integer coefficient polynomials are $EX$ learnable.

```
Algorithm Poly:  input x
If example contradicts last hypothesis
Then interpolate all examples
   and output the result
```

Example: target concept is $2x^3 - 2x^2 + 3$

| *Example* | *Hypothesis* |
|---|---|
| $f(0) = 3$ | $h(x) = 3$ |
| $f(1) = 3$ | $h(x) = 3$ |
| $f(2) = 11$ | $h(x) = 4x^2 - 4x + 3$ |
| $f(3) = 39$ | $h(x) = 2x^3 - 2x^2 + 3$ |
| $f(4) = 99$ | $h(x) = 2x^3 - 2x^2 + 3$ |
| $f(5) = 203$ | $h(x) = 2x^3 - 2x^2 + 3$ |
| $\vdots$ | $\vdots$ |

Point out

- Ignore efficiency
- Example doesn't return *index*
- Algorithm never stops

# Facts about $EX$ Learning

$EX(M)$ is the set of functions which $M$ learns "syntactically"

$EX$ is the class of all $EX(M)$

- (G67) $REC \notin EX$
- If $C$ is effectively enumerable, $C \in EX$
- (BB75) $EX$ not closed under union

THM: $REC \notin EX$

PF: Assume $REC = EX(M)$, Build $f \in REC$ st $f \notin EX(M)$.

Exists $m, n, e_1, e_2$ st $f = \phi_e$, $e \neq e_i$, and

$$M(f \cdot 1^m) \downarrow e_1 \quad \text{where} \phi_{e_1} = f \cdot 1^\infty$$
$$M(f \cdot 1^m \cdot 0^n) \downarrow e_2 \quad \text{where} \phi_{e_2} = f \cdot 1^m \cdot 0^\infty$$

Let $f = f \cdot 1^m 0^n$ and repeat.

CLAIM: $f$ is recursive (search all $m, n$ pairs, watching for mind changes until enough of $f$ is built to compute

THM: $C$ is effectively enumerable implies $C \in EX$

PF: Let $C = \{\phi_{f(0)}, \phi_{f(1)}, \cdots\}$.

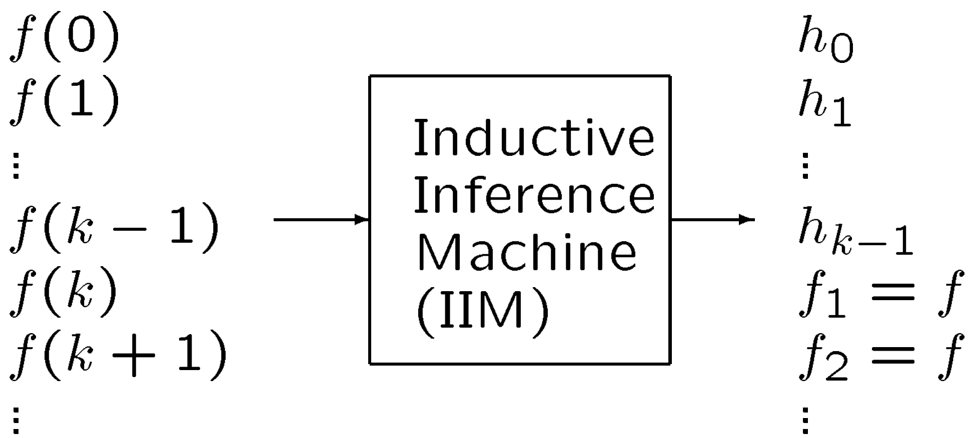
```
Algorithm for C

If current eg inconsistent with last hypothesis
Then Find least i st φ_f(i) consistent/w egs
   Output program for φ_f(i)

Example:  Algorithm for polynomials

If current eg inconsistent with last hypothesis
Then for i := ∞ Do
   Find least poly p with
      sum of coefficients and exps = i
   consistent with input egs
   If found, output p and leave loop
```

THM: EX not closed under union
PF: $EX^0$ and $EX^{=1} \subseteq EX$, $EX^1 \not\subseteq EX$

CLAIM: $EX^{=1} \subseteq EX$
PF: For each eg sequence, guess where anomaly is and guess correct patch. One pair of guesses will be correct.

CLAIM: $EX^1 \not\subseteq EX$
PF: (Later, see “Anomalies” slide)

# Semantic (Behaviorally Correct) Learning




For all $x$, $f(x) = f_1(x) = f_2(x) = \cdots$

# Facts about $BC$ Learning

$BC(M)$ the set of functions which $M$ learns "semantically"

$BC$ the class of all $BC(M)$

- (B74) $EX \subset BC$
- (G67) $REC \notin BC$
- (CS83) $BC$ not closed under union

THM: $REC \notin BC$—same as in EX

THM: $EX \subset BC$

PF: Let $C = \{f : \phi_{f(0)} =^1 f\}i$, $C \notin EX$ (see anomalies)

On input $\sigma$, output program for $\phi_{f(0)}$ with $\sigma$ patched into first $|\sigma|$ values. $BC$ learns input.

THM: BC not closed under union

PF: Let $SD = \{f : \phi_{f(0)} = f\}$ and $FS = \{f : f = 0\text{almost everywhere}\}$.

CLAIM: $SD \in BC$: output first input and halt.

CLAIM: $FS \in BC$: On input $\sigma$, output program for $\sigma 0^\infty$.

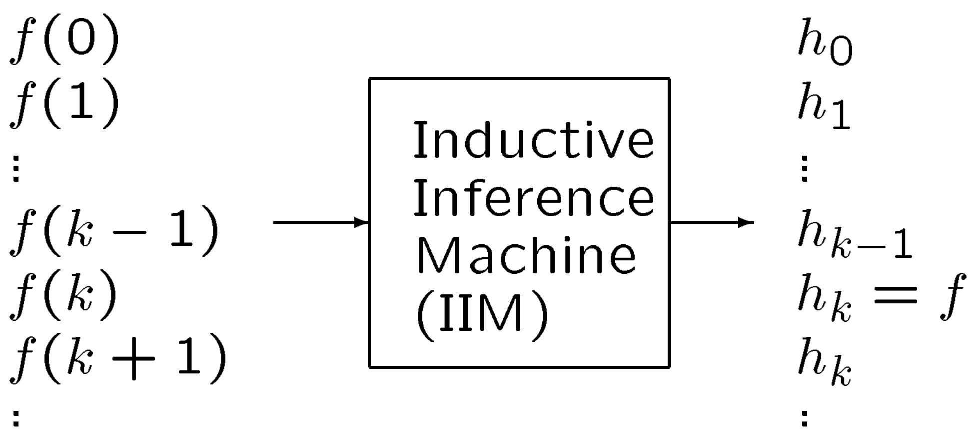
CLAIM: $FS\bigcup \notin BC$

PF: idea: use threat of finite support to guarantee mind change or error.

Build $f$ st $\phi_{f(0)} = f$ (by recursion theorem) as follows: Fix $M$. Given $\sigma$ so far, there is some $n$ such that either $M(\sigma 0^n)\downarrow$ or $\phi_{M(\sigma)} \neq \phi_{M(\sigma 0^n)}$. If not, then $\phi_{M(\sigma 0^m)} \in FS$ for some $m$ and so let $f(x) = \phi_{M(\sigma 0^m)}(x)$ for $x \leq |\sigma| + n$ and $f(x) = 1$ otherwise. In the first case, diagonalize, in the second, force a mind change.

In any case, there is an $f$ as desired.

## Making Mistakes




$h_k$ differs from $f$ on $\leq a$ values

## Facts about Anomalies

$f =^a g$ when $f$ and $g$ differ on at most $a$ values

$EX^a(M)$ are functions which $M$ $EX$-learns with $\leq a$ anomalies

$EX^a$ the class of all $EX^a(M)$

- (CS83) $EX = EX^0 \subset EX^1 \subset \cdots EX^*$

- (CS83) $BC = BC^0 \subset BC^1 \subset \cdots BC^*$

- (CS83) $REC \in BC^*$

THM: $C = \{f : \phi_{f(0)} \in EX^1 - EX$ PF: $C \in EX^1$ has been shown: patch $\phi_{f(0)}$ with egs so far to get guess.

Fix $M$ and show there is a $f \in C$ st $M$ doesn't learn $f$. Move anomaly marker. Let $\sigma = e\tau\diamond 0^n$ be $f$ constructed so far (by recursion theorem).

**Diagonalize** If $\phi_{M(\tau)}(\diamond) \downarrow y$, then replace $\diamond$ with a value which causes $f(\diamond) \downarrow\neq \phi_{M(\tau)}$. Move $\diamond$ to end.

**Mind Change** If $M(\sigma) \downarrow\neq M(\sigma\diamond)$ for some value of $\diamond$, then replace $\diamond$ with that value Move $\diamond$ to end.

**Extend** Let $\sigma = \sigma 0$.

Replace $\diamond$ with 0.

If the marker moved io, any machine on this sequence will either be wrong, or fail to converge. Else, any machine will err where the marker came to rest.

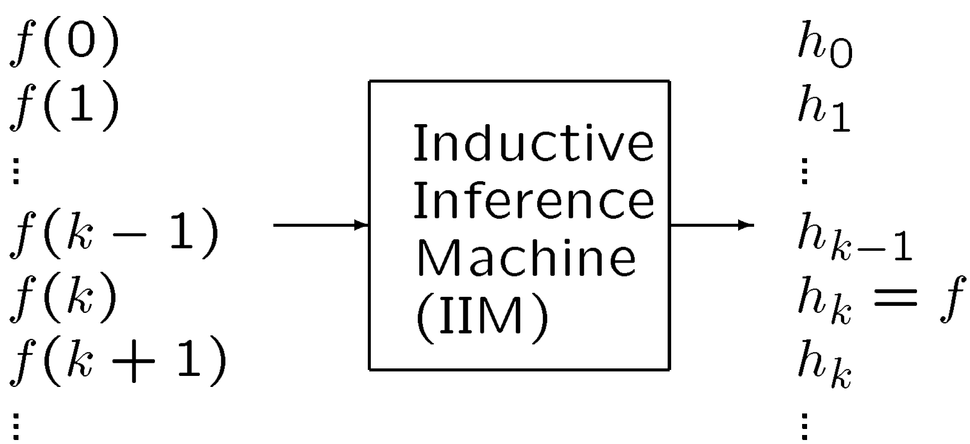
Properness of BC and EX hierarchies follows by adding more anomalies to class. EG, $C = \{f : \phi_{f(0)} =^k\} \in EX^k - EX^{k-1}$.

THM: $REC \in BC^*$

PF: To describe the appropriate $M$, describe $\phi_{M(\sigma)}(x) = f(x)$ for any $f \in REC$:

```
Input x
On training sequence σ
   Run the first |σ functions on 0...|σ|
      for x steps
   For each such function g consistent with σ
      compute g(x)
      output first converging answer
```

# Changing Your Mind




Has $k$ *mind changes*

# Facts about Mind Changes

$EX_m(M)$ are functions $M$ $EX$-learns with at most $m$ mind changes

$EX_m$ is class of all $EX_m(M)$

$EX_*(M)$ are functions $M$ $EX$-learns with finite mind changes

- (CS83) $EX_m \subset EX_{m+1}$
- (CS83) $BC_m = EX_m$
- (CS83) $EX_m^0 \subset EX_m^1 \subset \cdots EX_m^*$

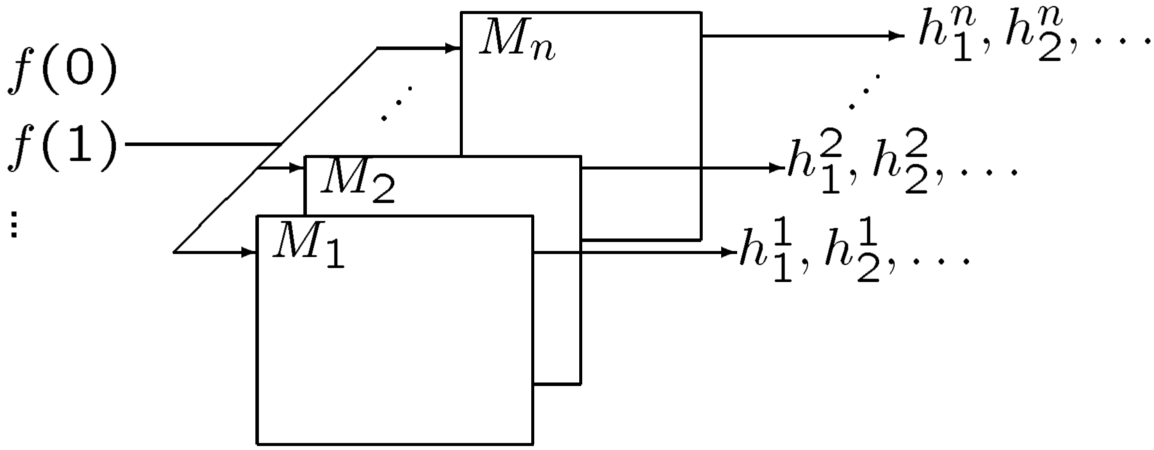
THM: $EX_m \subset EX_{m+1}$
PF: Show $m = 0$. Use functions which equal zero everywhere except at a single *accumulation point*. If $M(0^n) \downarrow e$, then $M$ misses the function st $f(n+1) = 1$.

THM: $BC_m = EX_m$
PF: Choose an IIM, count mind changes. After $m$, if that happens, stop changing mind. This is an $EX$ algorithm with $\leq m$ mcs.

THM: MC hierarchy proper with fixed anomalies
PF: same proofs carry over

NOTE: this means *mind changes are powerful*—they conflate syntax and semantics.

## Being a Team Player




Team learns $f$ if one machine does

## Facts about Team Learning

$[m,n]EX_b^a$ Classes of functions EX learned by $m$ out of $n$ team members with $a$ anomalies and $b$ mind changes

- (JS90) $[m,n]EX \subset [m,n+1]EX \subset \cdots$
- (S82) $m$ $EX^a$ machines can simulate $n$ $EX^b$ machine iff $n \leq m\left(1+\left\lfloor\frac{b}{a+1}\right\rfloor\right)$
- (D83) Ditto for $BC$ teams
- (JCN??) Ditto for Popperian machines

# More Facts about Team Learning

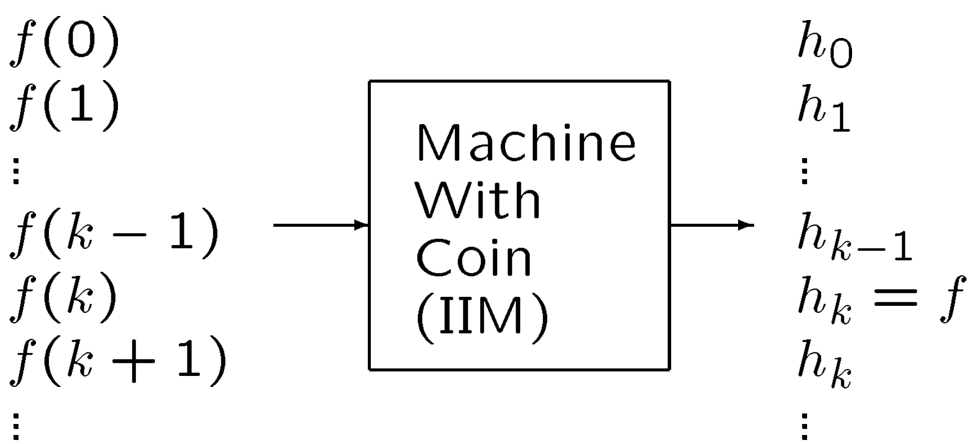
$[m,n]EX_b^a$ Classes of functions EX learned by $m$ out of $n$ team members with $a$ anomalies and $b$ mind changes

- (DPV91) $[1,2]EX_0 = [3,6]EX_0 \subset \cdots$
- (DPV91) $[2,4]EX_0 = [4,8]EX_0 \subset \cdots$
- (DPV91) $[2,4]EX_0 - [1,2]EX_0 \neq \emptyset$
- (BDV92) if $24/49 < a/b \leq 1/2$ then $[a,b]EX_0 = [1,2]EX_0$ and $[24,49]EX_0 - [1,2]EX_0 \neq \emptyset$

Accumulation points (as for mind changes) works for teams.

Odd thresholds I don't understand.

# Probabilistic Learning




Accepts with probability over coin tosses

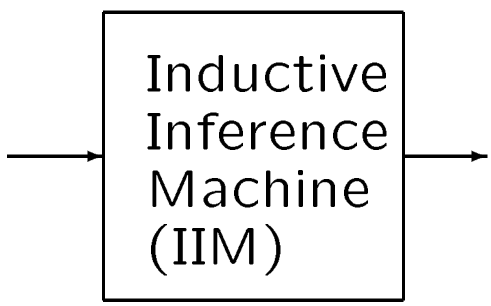
# Facts about Probabilistic Learning

$EX[p]$ Classes EX learned with probability $p$

- (P85) If $\frac{1}{n+1} < p \leq \frac{1}{n}$ then $EX[p] = [1, n]EX$

# Learning Formal Languages

$\langle w_0, b_0 \rangle$
$\langle w_1, b_2 \rangle$
$\vdots$
$\langle w_{k-1}, b_{k-1} \rangle$
$\langle w_k, b_k \rangle$
$\langle w_{k+1}, b_{k+1} \rangle$
$\vdots$




$gram_0$
$gram_1$
$\vdots$
$gram_{k-1}$
$gram_k$
$gram_k$
$\vdots$

$gram_k$ is grammar for target language $L$ and

$$b_i = L(w_i)$$

## Facts about Language Learning

- (Angluin) Context sensitive grammars can be learned
- But not from positive examples alone

- Other “Classical” Models
  - Query inferencing machines
  - Oracle inferencing machines
  - Ordinal inferencing machines
- Other “Efficient” Models
  - Probably approximately correct (PAC)
  - Membership queries
  - Equivalence queries

## Facts About Efficient Models

- (V85) Monomials are PAC learnable
- (V85) CNF is PAC learnable
- DFAs are learnable (exactly) with membership and equivalence queries

## Conclusions

- Inductive inferencing can be studied rigorously
- Common learning assumptions can be modelled mathematically
- "Common sense" and the truth are orthogonal
- Computational limitations imply learning limitations