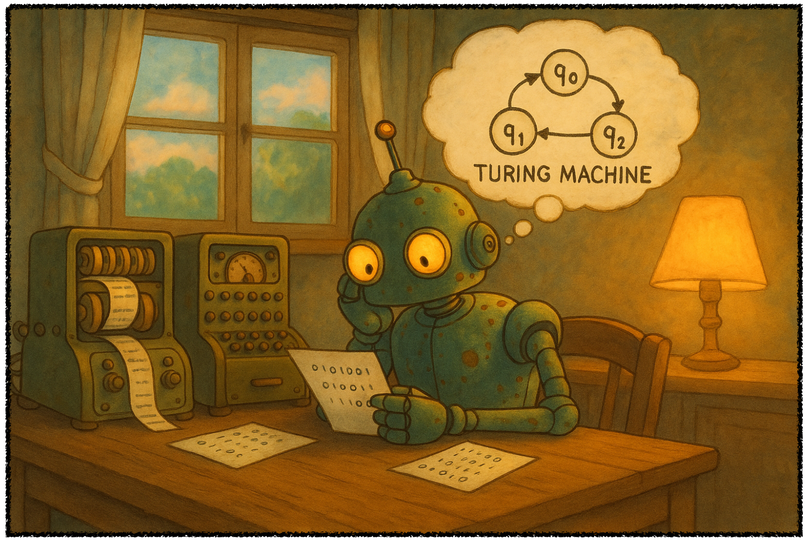


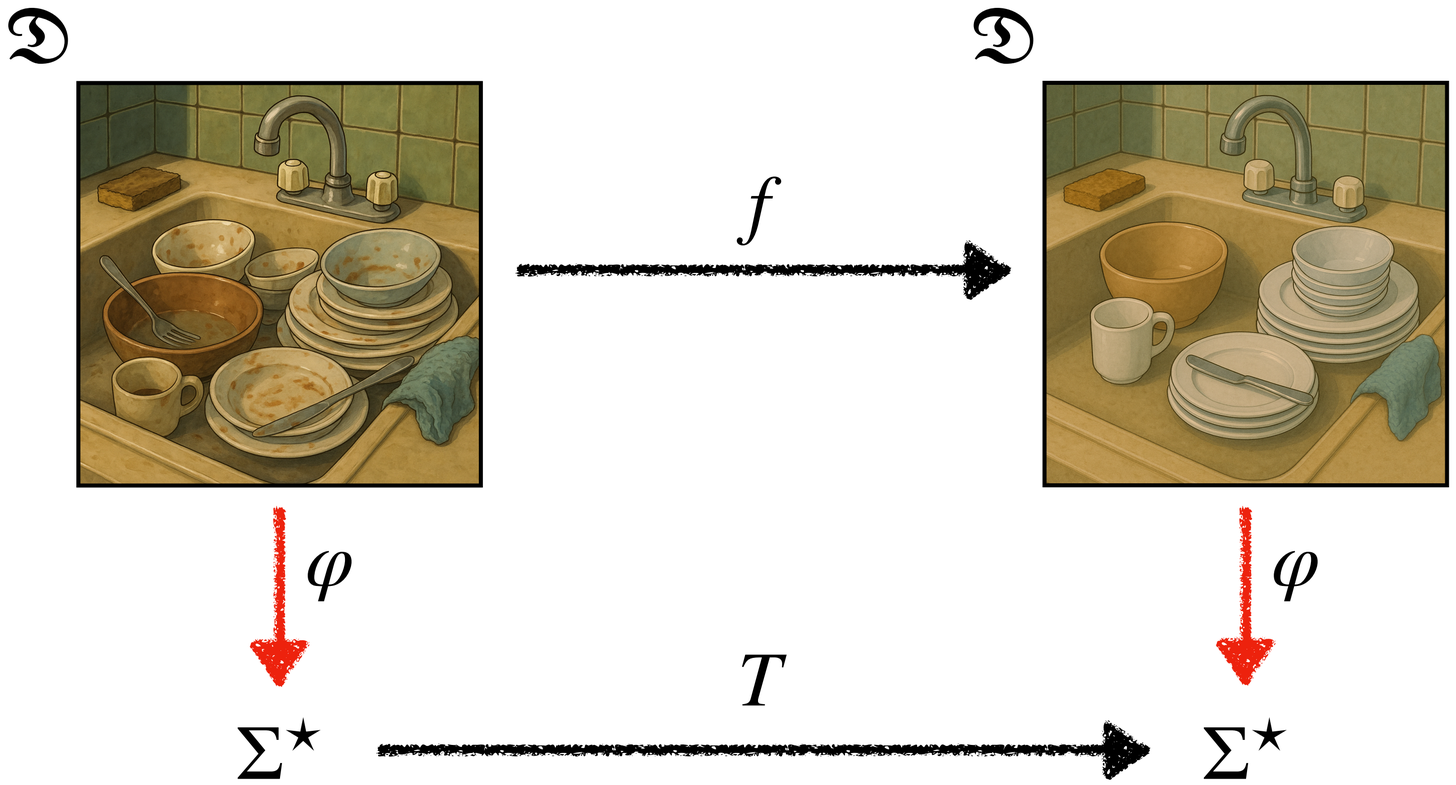
Learning Algorithms in the Limit

Hristo Papazov, Nicolas Flammarion

EPFL

COLT

Can an algorithm learn any computable function after finitely many observations?

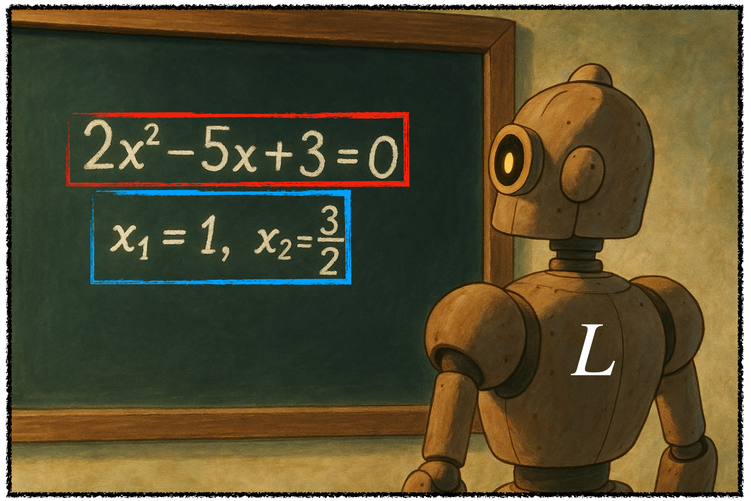


The Church-Turing Thesis

$$f = \varphi^{-1} \circ T \circ \varphi$$

$$C_{\Sigma^\star} = \{f : \Sigma^\star \to \Sigma^\star \,|\, f - \text{recursive}\}$$

Input-Output Observations



Learning in the Limit
(Gold, 1967)

Computation domain $\Sigma^\star$

Target function class $\Omega \subseteq C_{\Sigma^\star}$

Examples $\eta_f(x) = (x, f(x))$ from a ground-truth $f \in \Omega$

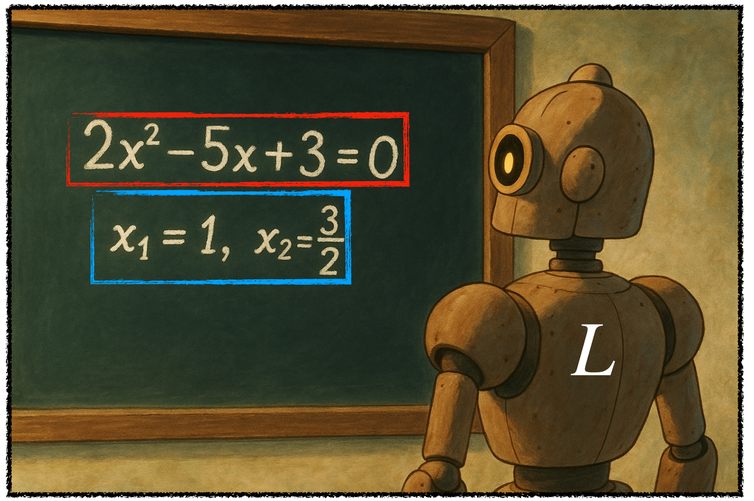
Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$

Hypothesis $h_t = L(\eta_f(w_1), \ldots, \eta_f(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

Learning in the Limit
(Gold, 1967)

Input-Output Observations



- **Criterion:** L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.
- **Theorem (Gold, '67).** The function class $C_{\Sigma^\star}$ is not learnable in the limit.

Learning systems observe computational processes, not the mere results of computation.

Learning with Computational Observations

$\mathscr{M}_{\Sigma^\star}$ — the family of deterministic discrete-time computation models on $\Sigma^\star$

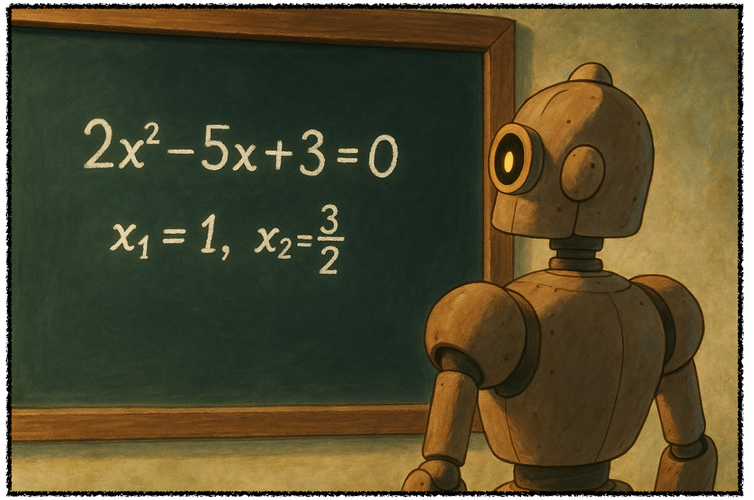
$F_{\mathscr{M}} \subseteq C_{\Sigma^\star}$ — the set of functions computed by the model class $\mathscr{M} \subseteq \mathscr{M}_{\Sigma^\star}$

f_M — the function computed by the model $M \in \mathscr{M}_{\Sigma^\star}$

$\alpha(M, x)$ — additional information about the computation $x \mapsto f_M(x)$

The $(\mathscr{M}, \alpha)$—Learning Problem

Input-Output Observations



Computation domain $\Sigma^\star$

Target function class $\Omega \subset C_{\Sigma^\star}$

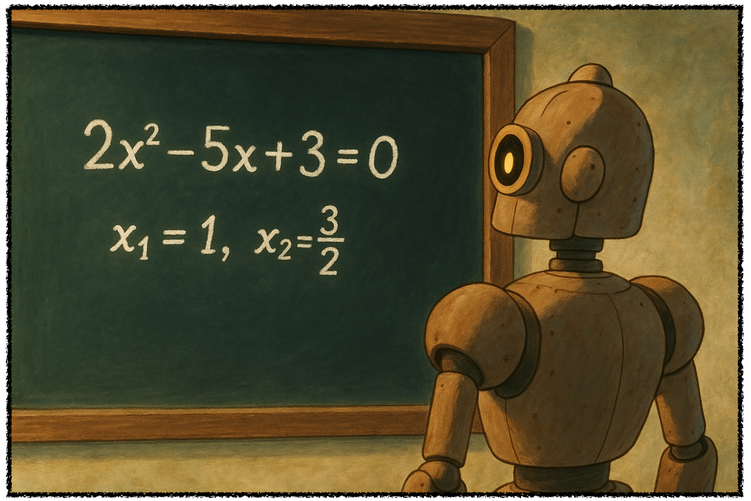
Examples $\eta_f(x) = (x, f(x))$ from a ground-truth $f \in \Omega$

Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$

Hypothesis $h_t = L(\eta_f(w_1), \ldots, \eta_f(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

Input-Output Observations



The $(\mathscr{M}, \alpha)$—Learning Problem

Computation domain $\Sigma^\star$ with input restriction $I \subseteq \Sigma^\star$

Target function class $\Omega \subset C_{\Sigma^\star}$

Examples $\eta_f(x) = (x, f(x))$ from a ground-truth $f \in \Omega$

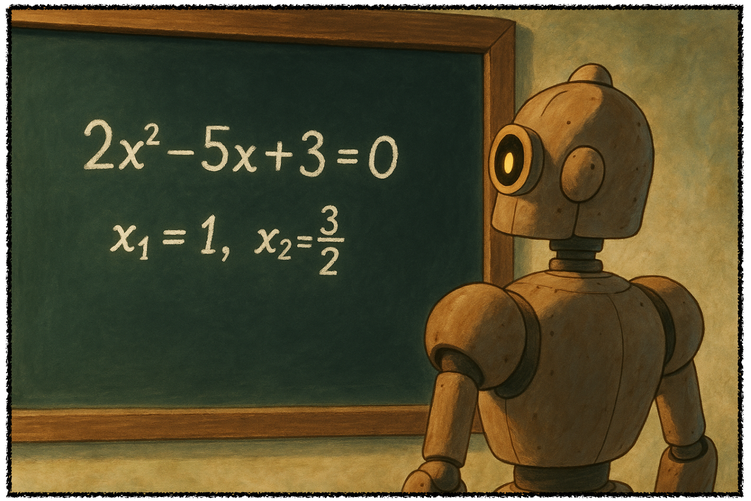
Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$

Hypothesis $h_t = L(\eta_f(w_1), \ldots, \eta_f(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

The $(\mathscr{M}, \alpha)$—Learning Problem

Input-Output Observations



Computation domain $\Sigma^\star$ with input restriction $I \subseteq \Sigma^\star$

Target model class $\mathscr{M} \subseteq \mathscr{M}_{\Sigma^\star}$

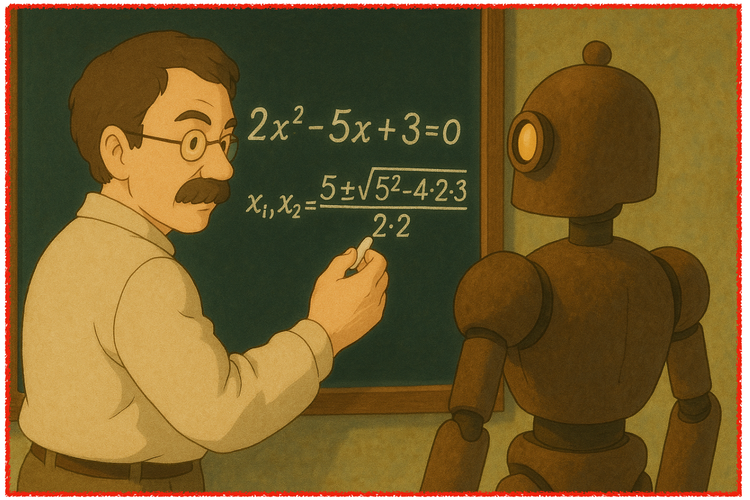
Examples $\eta_f(x) = (x, f(x))$ from a ground-truth $f \in \Omega$

Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$

Hypothesis $h_t = L(\eta_f(w_1), \ldots, \eta_f(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

The $(\mathscr{M}, \alpha)$—Learning Problem



Computation domain $\Sigma^\star$ with input restriction $I \subseteq \Sigma^\star$

Target model class $\mathscr{M} \subseteq \mathscr{M}_{\Sigma^\star}$

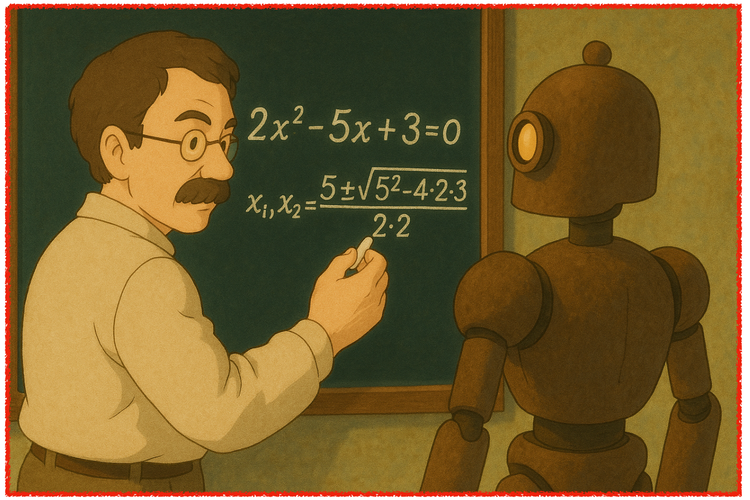
α-Examples $\eta_M(x) = (x, f_M(x), \alpha(M, x))$ from $M \in \mathscr{M}$

Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$

Hypothesis $h_t = L(\eta_f(w_1), \ldots, \eta_f(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

The $(\mathscr{M}, \alpha)$—Learning Problem



Computation domain $\Sigma^\star$ with input restriction $I \subseteq \Sigma^\star$

Target model class $\mathscr{M} \subseteq \mathscr{M}_{\Sigma^\star}$

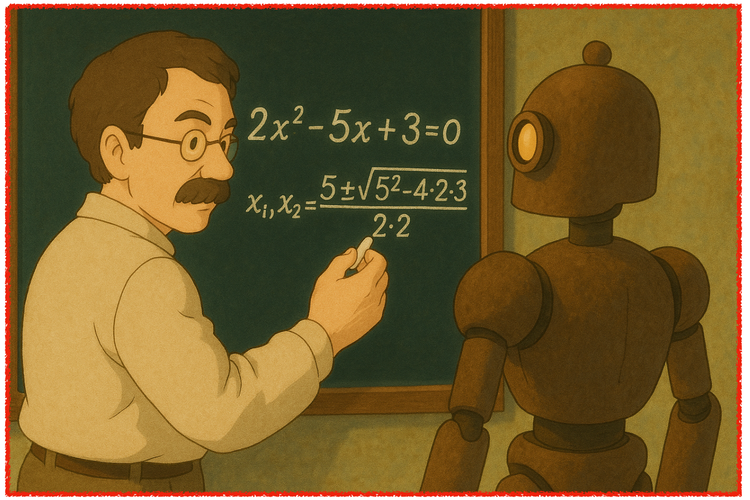
α-Examples $\eta_M(x) = (x, f_M(x), \alpha(M, x))$ from $M \in \mathscr{M}$

Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow I$

Hypothesis $h_t = L(\eta_f(w_1), \ldots, \eta_f(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

The $(\mathscr{M}, \alpha)$—Learning Problem



Computation domain $\Sigma^\star$ with input restriction $I \subseteq \Sigma^\star$

Target model class $\mathscr{M} \subseteq \mathscr{M}_{\Sigma^\star}$

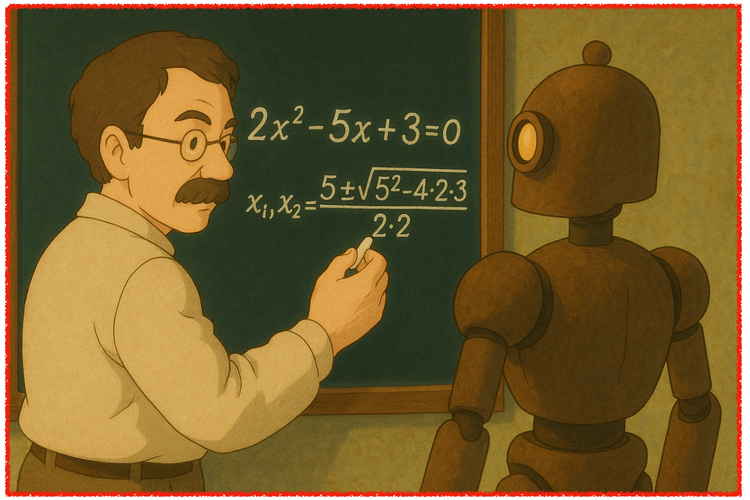
α-Examples $\eta_M(x) = (x, f_M(x), \alpha(M, x))$ from $M \in \mathscr{M}$

Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow I$

Hypothesis $h_t = L(\eta_M(w_1), \ldots, \eta_M(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L learns Ω in the limit if for every $f \in \Omega$ and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow \Sigma^\star$, there exists a time $t^\star(f, w)$ after which L outputs f.

The $(\mathscr{M}, \alpha)$—Learning Problem



Computation domain $\Sigma^\star$ with input restriction $I \subseteq \Sigma^\star$

Target model class $\mathscr{M} \subseteq \mathscr{M}_{\Sigma^\star}$

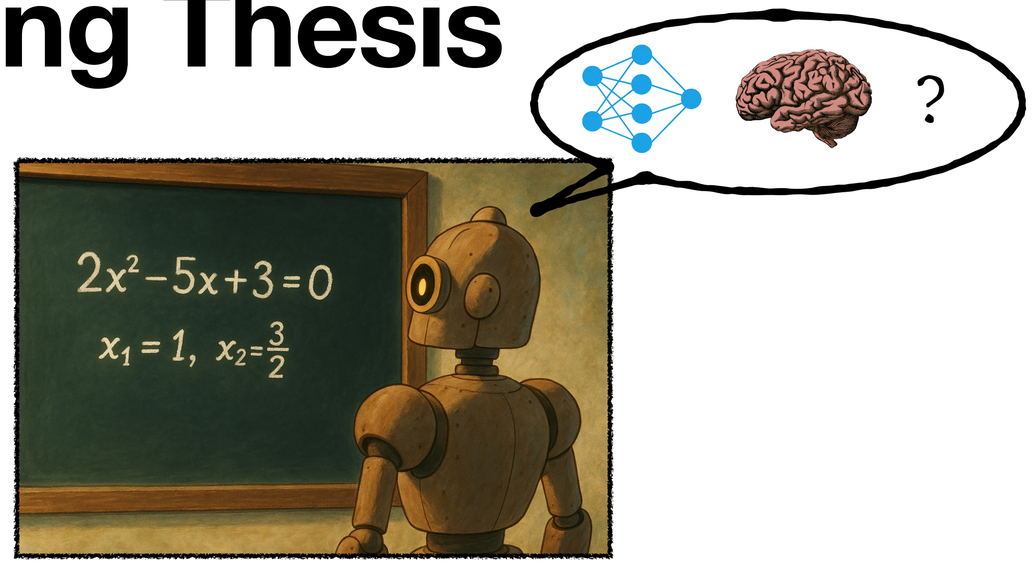
α-Examples $\eta_M(x) = (x, f_M(x), \alpha(M, x))$ from $M \in \mathscr{M}$

Exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow I$

Hypothesis $h_t = L(\eta_M(w_1), \ldots, \eta_M(w_t)) : \Sigma^\star \rightharpoonup \Sigma^\star$

Criterion: L solves the $(\mathscr{M}, \alpha)$—LP if for every $M \in \mathscr{M}$, every restriction $I \subseteq \Sigma^\star$, and every exhaustive input ordering $w : \mathbb{N} \twoheadrightarrow I$, there exists a time $t^\star(M, I, w)$ after which L outputs an I-equivalent representation of f: i.e., $\forall t \geq t^\star,\ h_{t|I} = f_{|I}$.

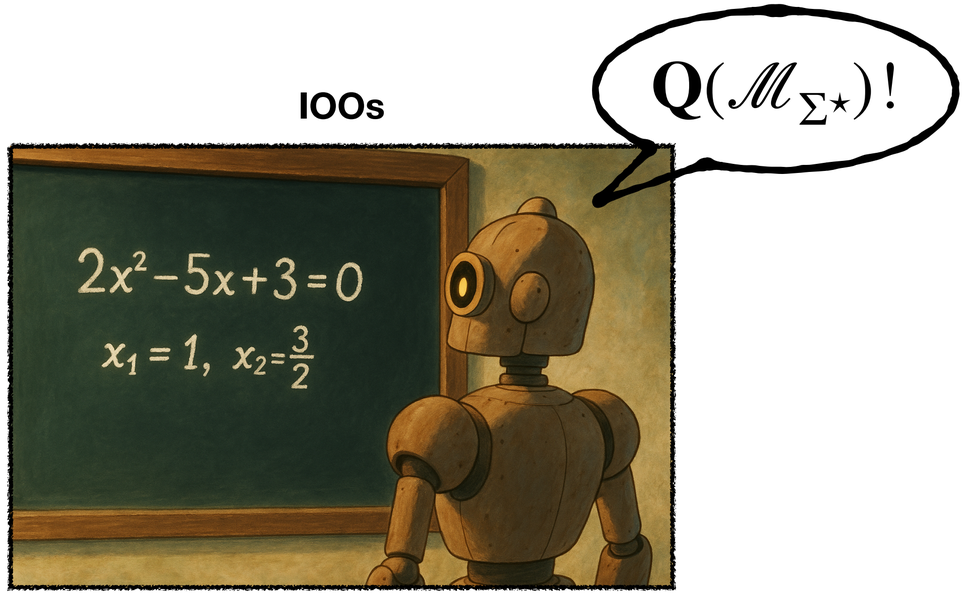
The Extended Church-Turing Thesis



$\mathscr{M}_{\Sigma^\star}$ contains an enormous variety of models.

(q-ECTT): Let $q : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be computable and monotonically increasing. Then, for any $M \in \mathscr{M}_{\Sigma^\star}$, there exists a Turing machine T operating on $\Sigma^\star$ and a constant $c \in \mathbb{N}$ s.t. T simulates n steps of M in at most $q(c, n)$ steps.

Input-Output Observations



$\alpha(M,x)=\varepsilon \Rightarrow$ We consider $(\mathscr{M},\varepsilon)\equiv F_{\mathscr{M}}$—LP.

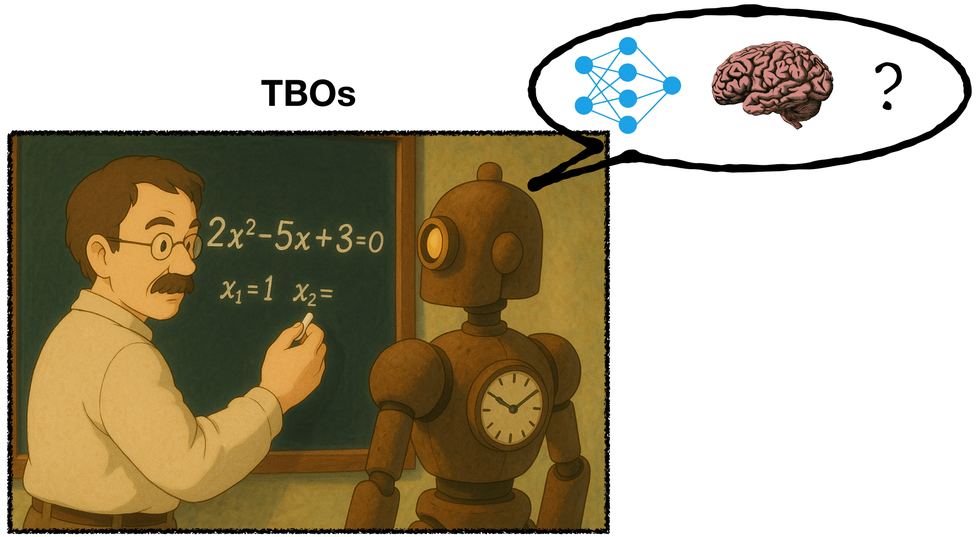
(Gold, '67): No algorithm solves the $C_{\Sigma^\star}$—LP.

For $\mathscr{M}\subseteq\mathscr{M}_{\Sigma^\star}$ and a computable monotonically increasing $Q:\mathbb{N}\times\mathbb{N}\to\mathbb{N}$, define the complexity class $\mathbf{Q}(\mathscr{M})=\cup_{c\in\mathbb{N}}\mathbf{TIME}_{\mathscr{M}}(Q(c,n))$.

Theorem (Time-Restricted IOOs). Assuming the q-ECTT, there exists an algorithm solving the $\mathbf{Q}(\mathscr{M}_{\Sigma^\star})$—LP.

Hence, popular complexity classes like $\mathbf{P}$, $\mathbf{NP}$, $\#\mathbf{P}$, $\mathbf{L}$, and $\mathbf{EXP}$ become learnable in the limit from restricted input sources.

Time-Bound Observations



- $\alpha_{\mathrm{TB}}(M, x) \in \mathbb{N}$ roughly upper-bounds the number of steps $M(x)$ takes.
- For each $M \in \mathscr{M}_{\Sigma^\star}$, there exists a constant c_M such that $t_M(x) \leq c_M \cdot \alpha_{\mathrm{TB}}(M, x),\ \forall x \in \Sigma^\star$.
- **Theorem (Universal TBO-Learning).** Assuming the q-ECTT, there exists an algorithm solving the $(\mathscr{M}_{\Sigma^\star}, \alpha_{\mathrm{TB}})$—LP.

Conclusion

Summary:

- Introduced computational observations and restricted input sources
- Proved learnability of parametrized complexity classes from IOOs
- Proved learnability of all computable functions from TBOs

More in the paper:

- Observations of intermediate steps of computational agents
- Sample-efficiency arguments through characteristic sets

Future Directions:

- We can discuss!