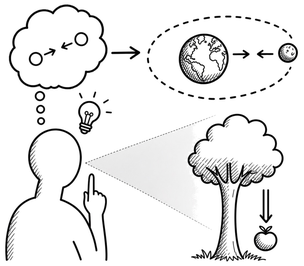
BabyTheorist: A Benchmark for Learning to Theorize the World from Observation Alone

Doojin Baek*, Junyeob Baek*, Mingyu Jo, Hosung Lee, Taegu Kang, Gyubin Lee, Sungjin Ahn *Equal contribution

Motivation

Can a model continually acquire theories of the world?



- Humans **acquire** theories by watching the world, then **apply** them to new situations
  - falling objects → gravity → orbits
- ARC-AGI, Bongard, and Raven **group tasks by one hidden rule**: they test **application**.
- Acquiring rules from **ungrouped** observations is left outside the evaluation.

From a continual stream of observations, a theory system must keep evolving.

Learning to Theorize (L2T)

Building on L2T (Baek et al., ICML 2026), each **phenomenon** is a before-after pair (x, y) produced by a hidden **program**: an ordered sequence of **primitives** never shown to the learner.

$$\underbrace{y}_{\text{observed}} = \underbrace{f_{z_K} \circ \cdots \circ f_{z_1}}_{\text{hidden program } \tau} \underbrace{(x)}_{\text{observed}}$$

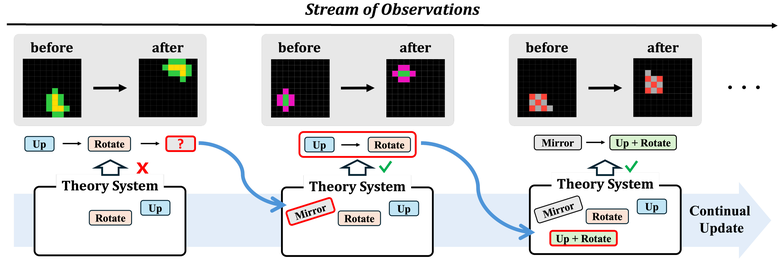
No program, action, or task labels are given; the learner sees **only** (x, y).

To theorize, a learner must

(i) Discover reusable primitives across phenomena

(ii) Compose them into structured, executable explanations

(iii) Explain novel phenomena by recombining them



A theory system grows: it discovers a new theory when existing ones fail (Mirror) and turns recurring compositions into reusable units (Up + Rotate) for later phenomena.

⇨ **Evaluation by transferability:** a program inferred from (x, y) must map a new input x' of the same program to its y', which memorization cannot do.

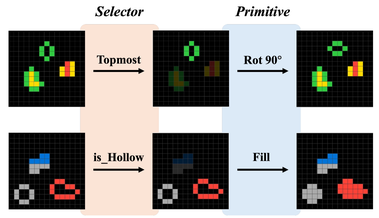
Baek et al., "Learning to Theorize the World from Observation," ICML 2026.

BabyTheorist: 3 Core Designs

BabyTheorist extends L2T to a **continual curriculum** of grid observations, built on **three design controls**; learners see only **ungrouped before-after pairs**.

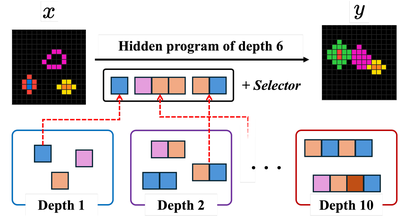
① Hidden program language

- Each program step pairs a **selector** (*what*) with a **primitive** (*how*).
  ⇨ The language is hidden: learners must **discover its primitives**.



② Frequent subprograms

- Some primitive sequences recur, both alone and inside longer programs.
  ⇨ Compressing them into **reusable abstractions** shrinks the search space.

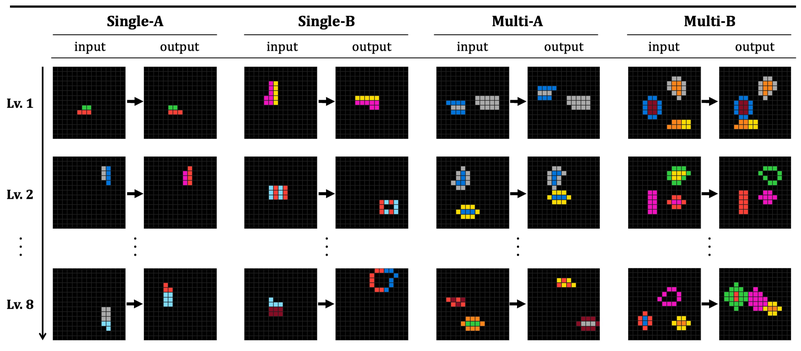


③ Level-based curriculum

- Levels shift toward longer programs while still mixing in shorter ones.
  ⇨ Early theories become **building blocks** for longer programs.

Four benchmark modes

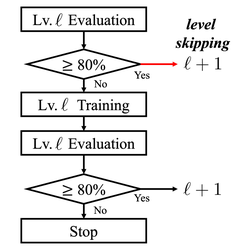
- **Single / Multi:** one object vs. several objects with relational primitives.
- **A / B:** fixed vocabulary vs. new primitives added as levels progress.



Mode	# Objects	Selectors	# Primitives	Primitive Expansion	Program space
Single-A	1	–	8	–	8^d
Single-B	1	–	11	✓(8 → 11)	11^d
Multi-A	2	by_index	11	–	22^d
Multi-B	3	all selectors	14	✓(11 → 14)	216^d

Continual Train-Test Protocol

- **Entry evaluation:** before training on level ℓ; passing ($\geq 80\%$) **skips its training data** and moves to $\ell + 1$.
- **Training:** otherwise, learn from 250K ungrouped observations of level ℓ, with **no program labels**.
- **Level evaluation:** on 10K tasks; passing ($\geq 80\%$) advances to $\ell + 1$, while failing ends the curriculum.
- **Each task:** 2-5 support pairs of one hidden program, added until the query output is **unambiguous**.

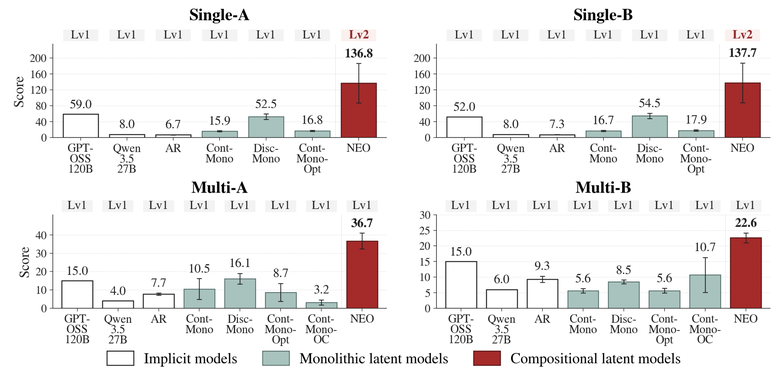


- **Score:** a_ℓ is the exact-match accuracy at the end of level ℓ, taken from the entry evaluation if the level is skipped.
- **Efficiency:** breaks ties, favoring early gains; S_t is the Score after t of $T_{\max} = 8$ stages, and S_{final} after the last.

$$\text{Score} = \sum_{\ell=1}^{L_{\text{reached}}} 100\, a_\ell \qquad \text{Eff} = \frac{1}{T_{\max} S_{\text{final}}} \sum_{t=1}^{T_{\max}} S_t$$

Results

- **All baselines stall within two of eight levels**; only NEO reaches Level 2 (Single).
- **LLMs fail Level 1** (GPT-5.6 Sol: 51-64%); **PPO with the true DSL** clears Single-A.



Analysis: three challenges for future methods

1. Discovering reusable primitives

⇨ **Favor reusable trans-formations from the start**

Cont-Mono fits every support yet misses the query in 23.5% of tasks; NEO, only 2.4%.

2. Learning where to apply

⇨ **Learn which object to change, not just how**

NEO applies the right change to the wrong object in 25.4% of depth-one Multi-A cases.

3. Explaining higher-depth phenomena

⇨ **Explore compositional inductive biases**

Models reproduce only part of a multi-step change, e.g., a rotation without the flip.

- Most of the curriculum remains unsolved by the evaluated baselines; going further calls for overcoming these three challenges.
- Correspondence: doojin.a.baek@kaist.ac.kr, wnsdlqjtm@kaist.ac.kr