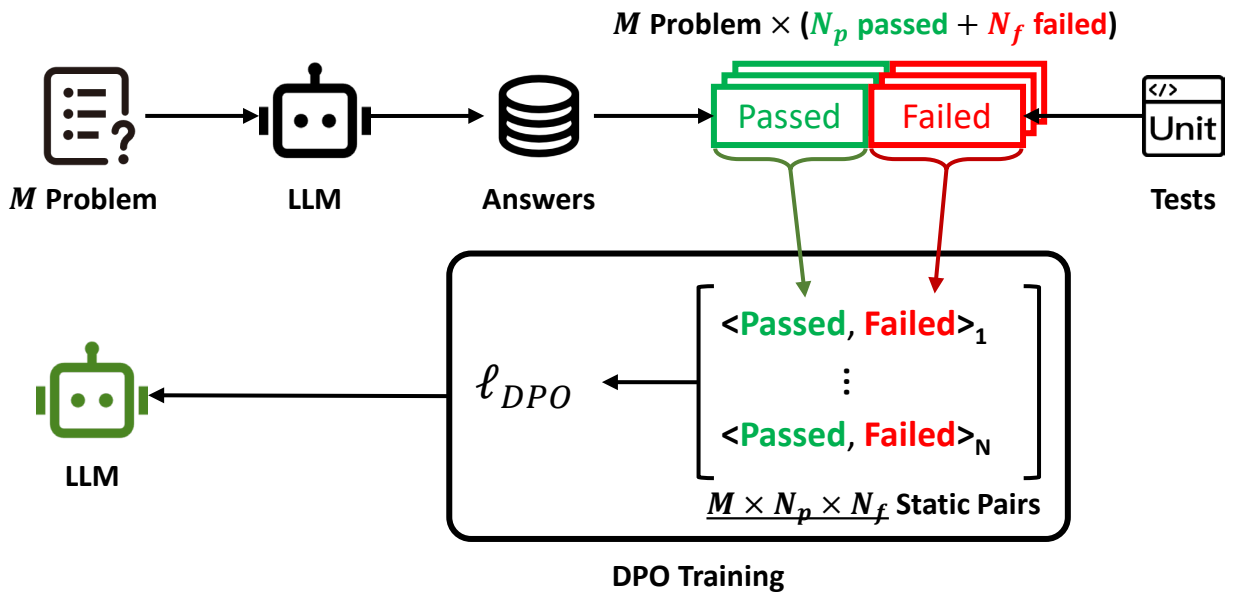
[Code]: AP2O (reducing code errors)

- AP2O: Adaptive Progressive Preference Optimization
  - **Problem:** LLM-generated code has **compilation or runtime errors**
  - **Goal:** Reduce code errors, improve **code quality** & **pass rate** & **acceptance rate**
  - **Solution:** Progressive preference optimization + adaptive error replay
  - **Results**: Improve **pass@k** by up to **3%** for **0.5B~34B LLM** on *EvalPlus*, *LiveCodeBench*, etc.
    - Qwen2.5-Coder, CodeLlama, DeepSeek-Coder, Qwen2.5, Llama3, Qwen3
  - **Bonus**: Reduce **data requirement** greatly (only using 4%~60% data)
  - **Findings**:
    - **Poor models**: Progressing from **low to high** error frequency
    - **Strong models**: Progressing from **high to low** error frequency

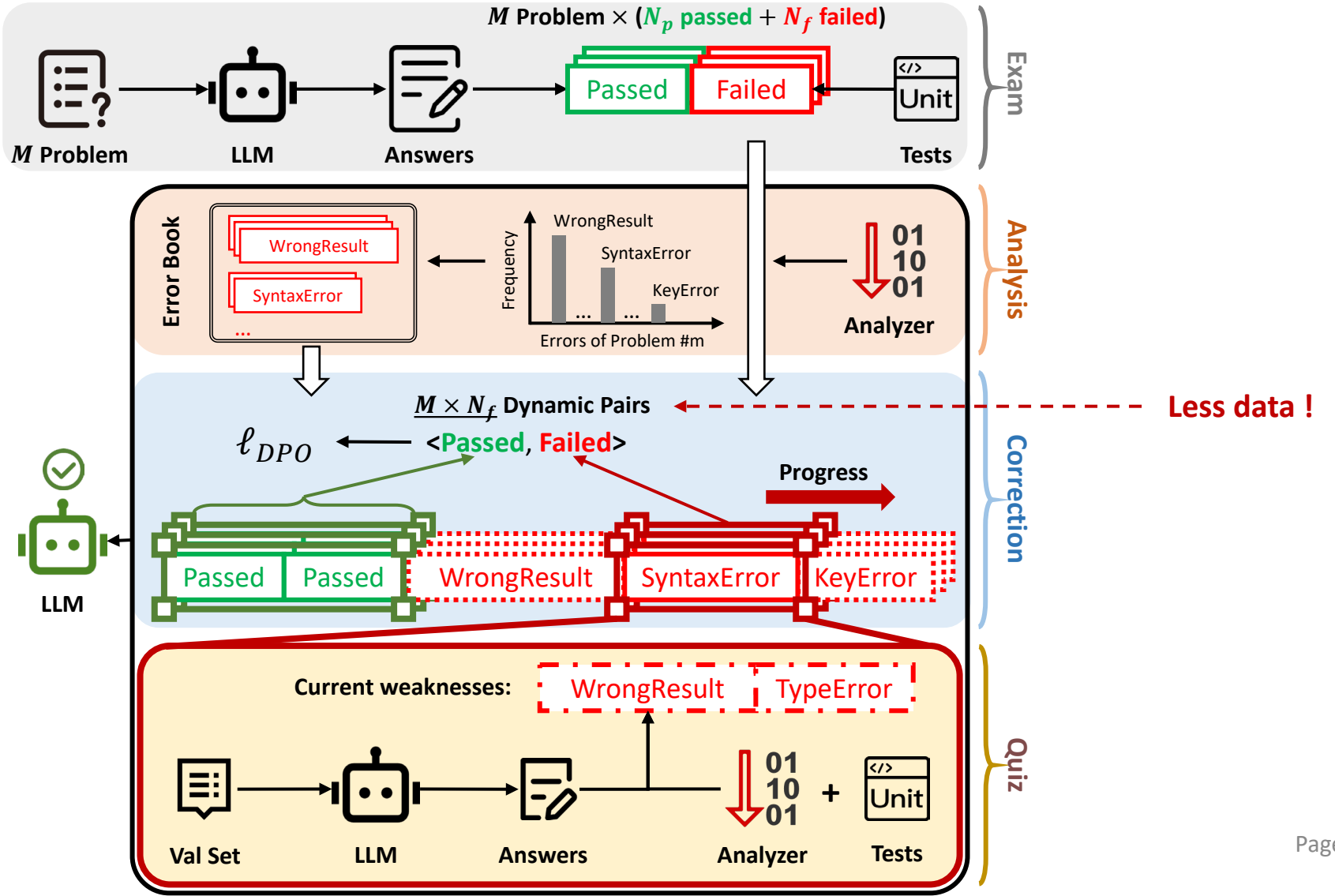
[Code]: Existing methods

- Traditional DPO training for code generation
  - **Problem 1:** Unawareness of **code errors**
  - **Problem 2:** Inability to **focus** on specific error types **(SyntaxError, TypeError, etc.)**



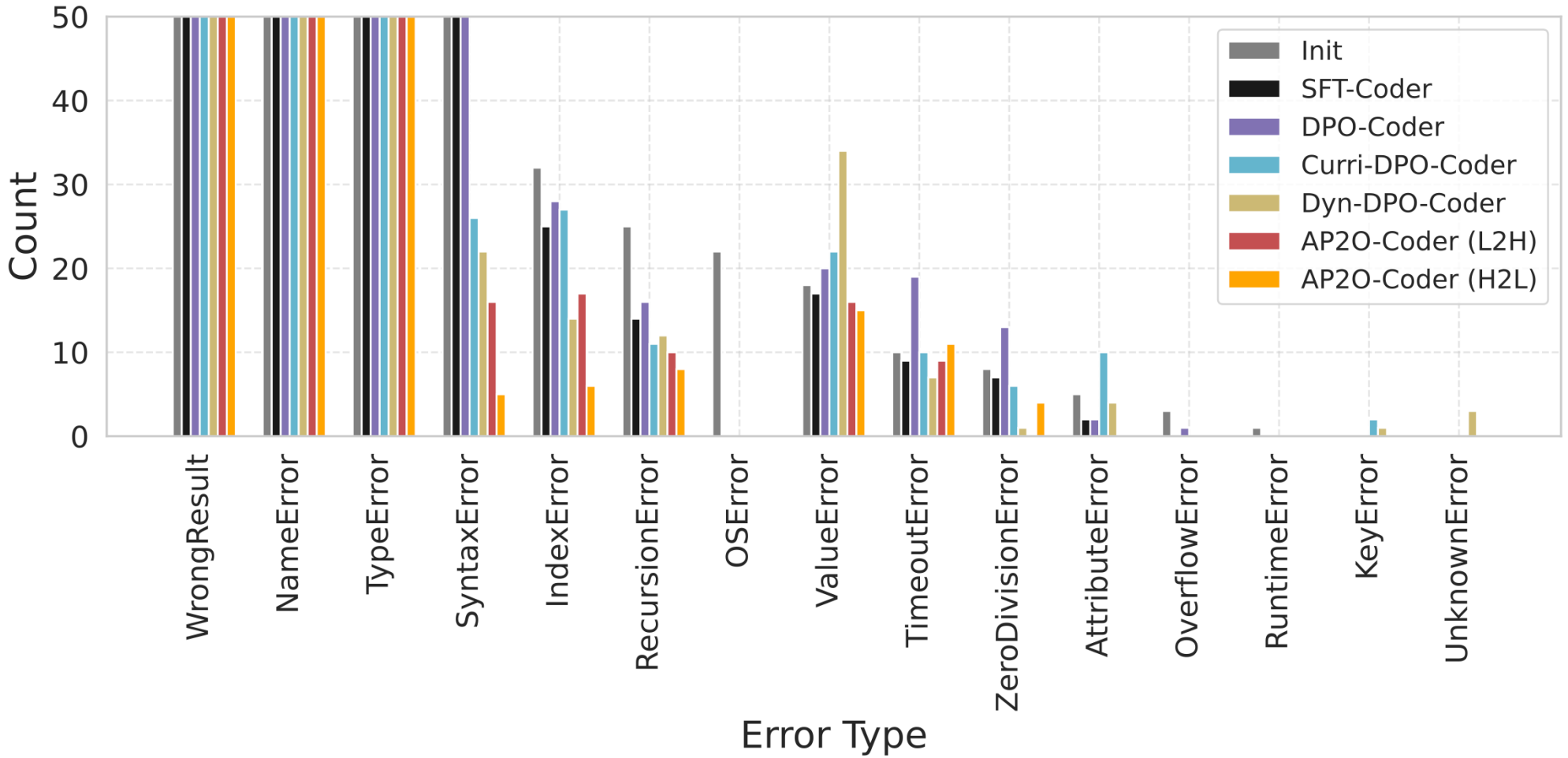
[Code]: AP2O (reducing code errors)

- **Solution:** Progressive preference optimization + adaptive error replay



[Code]: AP2O (reducing code errors)

- AP2O reduces code **errors**



Outperform baselines by up to 3% in pass@k

[Code]: AP2O (reducing code errors)

- AP2O (**H2L**) is better for large models.
  - **Concentration** then **exploration** fits **mature LLMs**
  - H: LLM sees **identical error types** in adjacent updating steps - **specification**
  - L: LLM sees **various error types** in adjacent updating steps - **generalization**

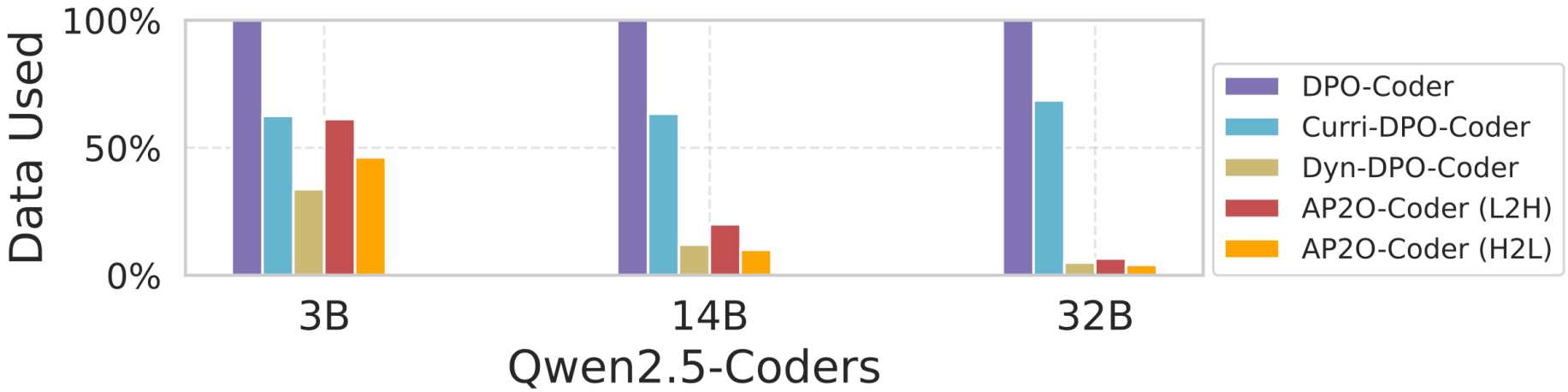
LLM Type	CodeLlama			DeepSeek-Coder			Qwen2.5-Coder					
LLM Size	7B	13B	34B	1.3B	6.7B	33B	0.5B	1.5B	3B	7B	14B	32B
Init	36.8	41.3	46.2	64.6	77.4	78.4	53.0	69.3	83.5	87.1	90.4	91.5
SFT-Coder	37.9	43.2	46.8	64.8	75.9	78.9	60.1	70.4	85.1	87.4	90.7	90.9
DPO-Coder	38.3	42.3	45.2	63.5	77.2	78.7	56.8	73.2	84.5	87.9	90.8	91.0
Curri-DPO-Coder	38.7	42.4	46.5	63.8	76.6	79.2	53.3	73.1	83.7	87.2	90.2	90.8
Dyn-DPO-Coder	38.6	42.3	44.9	63.4	76.2	78.8	57.1	71.5	84.7	87.6	90.7	91.6
AP2O-Coder (L2H)	**39.8**	43.1	47.9	**65.9**	77.6	79.1	**61.5**	**76.3**	85.7	88.1	90.8	91.8
AP2O-Coder (H2L)	38.9	**44.5**	**49.6**	64.7	**78.8**	**80.1**	56.5	71.7	**86.3**	**88.9**	**91.4**	**92.2**

Table 1: The $pass@1$ on EvalPlus (HumanEval) across various types and sizes of code LLMs.

Outperform baselines by up to 3% in pass@k

[Code]: AP2O (reducing code errors)

- AP2O is also sample **efficient** (only using 4%~60% data)



Outperform baselines by up to 3% in pass@k