On-Policy Optimization with Group Equivalent Preference for Multi-Programming Language Understanding

Haoyuan Wu[1*] Rui Ming[1*] Jilong Gao[1*] Hangyu Zhao[1] Xueyi Chen[1] Yikai Yang[1]
Haisheng Zheng[2] Zhuolun He[1,2] Bei Yu[1]
[1]The Chinese University of Hong Kong [2]ChatEDA Tech

The Coding Capability Gap in LLMs

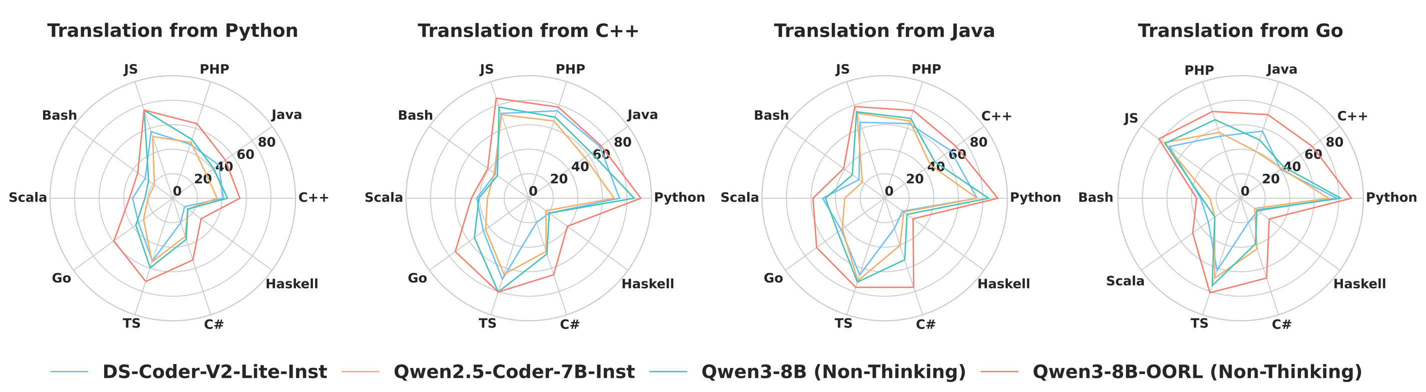

Figure 1. Code translation performance on our CrossPLEval benchmark reveals a significant capability gap between different programming languages for SOTA models.

Code Translation

- Code Translation among different high-level programming languages.
- Code Translation between high-level programming languages and low-level intermediate representations (IRs).

Contributions

- We introduce **OORL**, a novel RL framework integrating on-policy and off-policy strategies to bridge the language capability gap via **code translation tasks**.
- We develop **GEPO**, a novel preference optimization method that extends beyond pairwise comparisons by explicitly modeling functional equivalence within groups of IRs.
- Extensive evaluations show our method achieves superior performance across various code benchmarks in multiple programming languages, including low-resource ones.

OORL: A Hybrid RL Framework

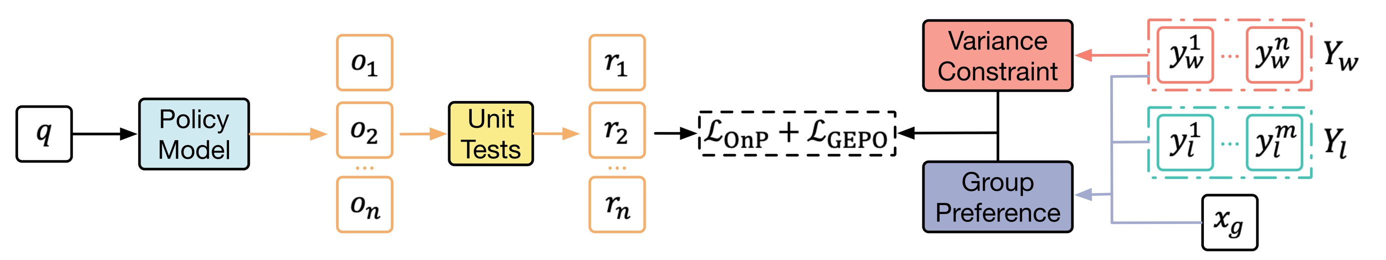

Figure 2. Overview of OORL, which combines on-policy RL with rule-based rewards for correctness and off-policy preference optimization (GEPO) for nuanced understanding.

On-Policy RL for Correctness

Guiding the model with a clear success signal.

- We train the LLM on **code translation tasks** (e.g., Python to Java).
- A simple, **binary rule-based reward** is used:

$$R_{\text{rule}}(q, o) = \begin{cases} 1, & \text{if translation is successful,} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

- This provides a strong but coarse-grained signal.

Off-Policy RL for Nuance

The binary reward is coarse-grained. It cannot distinguish between multiple valid implementations or provide process-level supervision on **functional equivalence**.

Our Solution: We use compiler Intermediate Representations (IRs), which are language-agnostic, to capture fine-grained code functionality.

Group Equivalent Preference Optimization

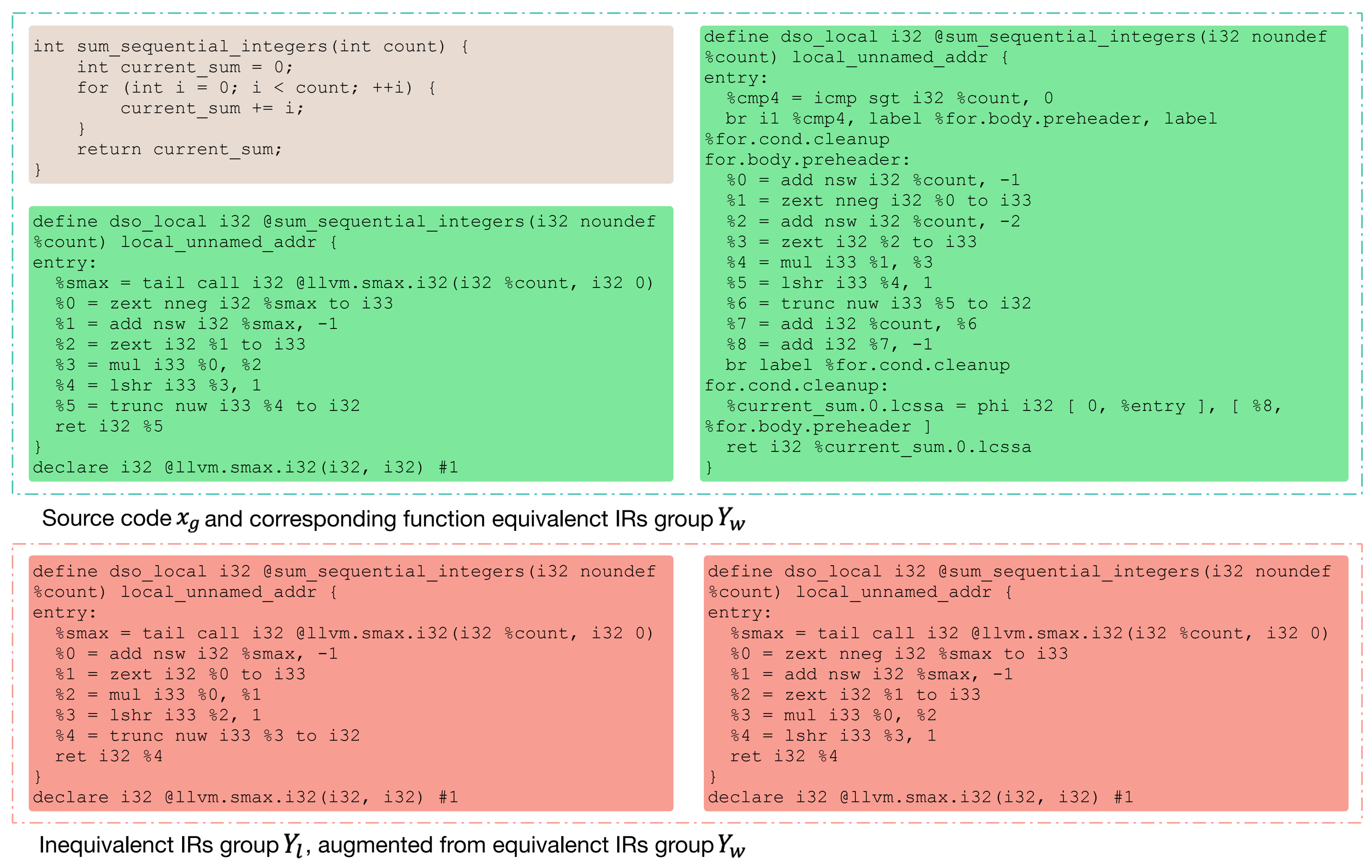

Figure 3. GEPO trains on groups of "winner" (functionally equivalent) IRs and "loser" (inequivalent) IRs.

The GEPO Loss Function

Enforcing Preference and Equivalence.

$$\mathcal{L}_{\text{GEPO}} = \underbrace{-\mathbb{E}[\log \sigma(\beta(\frac{1}{n}\Sigma \hat{r}_w - \frac{1}{m}\Sigma \hat{r}_l))]}_{\text{Preference: Winners > Losers}} + \underbrace{\lambda \mathbb{E}[\text{Var}(\hat{r}_w)]}_{\text{Equivalence: Penalize variance}}$$

- The first term pushes the model to prefer the winner group over the loser group.
- The second term encourages the model to view all IRs within the winner group as equally good.

Experiment Settings

- **Base Model**: Qwen3-8B
- **Training Strategy**: OORL (On-policy REINFORCE++ and off-policy GEPO)
- **Baselines**:
  - DS-Coder-V2-Lite-Inst, Qwen2.5-Coder-7B-Inst, Qwen3-8B
- **Benchmarks**:
  - **MultiPL-E**: Code generation (8 languages).
  - **CrossPLEval**: Code translation (11 languages).

Main Results: MultiPL-E (Code Generation)

Table 1. Pass@1 performance on MultiPL-E.

	Py	C++	Java	PHP	Bash	JS	TS	C#	Avg.
DS-Coder...	81.10	32.91	68.35	72.67	19.62	81.36	83.33	41.13	60.06
Qwen2.5...	88.40	57.14	70.88	73.91	43.67	78.88	84.27	52.53	68.71
Qwen3-8B	82.60	71.42	70.25	50.31	36.07	83.22	84.27	42.41	65.07
Ours	**90.06**	**83.23**	**83.54**	**78.26**	**46.20**	**89.44**	**85.53**	**54.22**	**76.31**

Main Results: CrossPLEval (Code Translation)

Table 2. Pass@1 performance on CrossPLEval.

	Model	Target Language											
		Python	C++	Java	PHP	JS	Bash	Scala	Go	TS	C#	Haskell	Avg.
Py	DS-Coder...	-	41.17	46.21	46.21	57.14	27.73	32.77	34.45	53.78	21.00	11.76	37.22
	Qwen2.5...	-	36.13	33.61	47.89	52.94	18.54	20.16	29.41	54.62	32.77	15.12	34.12
	Qwen3-8B	-	44.53	41.17	50.42	75.54	24.36	22.68	37.25	59.66	35.29	15.13	40.60
	Ours	-	**54.62**	**52.94**	**63.94**	**75.63**	**35.29**	**35.29**	**59.32**	**71.42**	**52.94**	**28.48**	**52.99**
C++	DS-Coder...	73.91	-	71.73	75.00	72.82	33.69	42.39	45.65	69.56	20.65	20.65	52.61
	Qwen2.5...	69.56	-	56.52	66.30	71.73	34.78	33.69	43.47	65.21	45.65	17.39	50.43
	Qwen3-8B	85.86	-	63.04	69.56	78.26	31.52	40.65	54.71	80.43	47.82	20.65	57.25
	Ours	**91.30**	-	**72.82**	**78.15**	**85.86**	**41.30**	**46.73**	**74.02**	**80.43**	**65.65**	**38.91**	**67.52**
Java	DS-Coder...	75.28	66.29	-	64.04	65.16	25.84	50.56	43.82	65.82	25.84	17.97	50.06
	Qwen2.5...	75.28	46.06	-	66.29	73.03	22.47	32.58	42.69	70.78	40.44	19.10	48.87
	Qwen3-8B	84.64	50.56	-	68.53	74.15	32.58	48.31	47.19	71.91	52.80	22.47	55.31
	Ours	**92.13**	**71.91**	-	**75.28**	**78.65**	**41.34**	**58.42**	**68.53**	**76.40**	**76.40**	**28.53**	**66.76**

Ablation Studies

Table 3. Contribution of each component in OORL.

Model	On-Policy	Off-Policy	MultiPL-E	CrossPLEval
Qwen3-8B (Base)	-	-	65.06	50.96
+ REINFORCE++	✓	-	73.60	58.46
+ REINFORCE++	✓	DPO	73.48	58.70
Ours (OORL)	✓	**GEPO**	**76.31**	**63.15**

- **On-policy RL** provides a major boost.
- **GEPO** provides significant additional gains over standard DPO, proving the value of modeling functional equivalence.

Conclusion

- We proposed **OORL**, a hybrid RL framework to improve the multi-language capabilities of LLMs via code translation.
- Our key innovation, **GEPO**, leverages groups of Intermediate Representations to teach the model about functional equivalence, providing a richer signal.
- Our model, **Qwen3-8B-OORL**, establishes a new state-of-the-art on code generation and translation benchmarks, generalizing even to low-resource languages.

Department of Computer Science and Engineering, The Chinese University of Hong Kong
byu@cse.cuhk.edu.hk