Expressive Type System is Savior of LLM

Dongjae Lee

2025.10.15

Abstract

While latest LLMs are good at generating safe code, inserting try-catch blocks and if-guards everywhere is not what we want [1]. However, getting LLMs to insert exception handling logic in the right places is very challenging. I believe highly expressive type systems can solve this problem.

Through the RLVR (Reinforcement Learning with Verifiable Reward) methodology, we have been able to create reasoning models with overwhelming performance in domains where automatic verification is possible. However, this is not a panacea. Depending on what rules are used to assign rewards, LLMs can exploit these rules, leading to "reward hacking."

From a safety perspective, reward hacking manifests as follows: LLMs insert exception handling code literally everywhere in order to generate safe code. While they have created safe code, it is not in the form we desired. Creating a perfect reward system is a challenge not only for reinforcement learning but for all of humanity (Goodhart's Law).

To solve this problem, I propose making LLMs effectively use expressive type systems. LLMs insert exception handling everywhere due to uncertainty about exception occurrence. In particular, even if they have previously determined the likelihood of exception occurrence, the LLMs' context is reset with each session. If types could indicate exception possibilities, LLMs could confidently write core logic without excessive safety code.

During the training process, if unnecessary exception handling is inserted, this could be detected through type checkers. Current LLMs cannot yet freely handle even Rust's type system. It is time to move beyond Python to Rust and Lean.

[1] [Super Safe Divison]