人工智能代码编辑器的未来方向

最近，我在 GitHub Actions 中添加了一個 `xelatex` 管道。

在 GitHub 流程中，我遇到了 `fontawesome5` 包的問題。4o-mini 提供的解決方案（安裝 TeX Live 2021 並使用 `tlmgr install fontawesome5`）對我無效。然而，4o 建議了一個更好的方法：升級到 TeX Live 2023 並仍然使用 `tlmgr` 來安裝 `fontawesome5`。雖然這並沒有完全解決問題，但切換到 TeX Live 2023 顯著改善了情況。

我使用了 ChatGPT 來幫助找出問題。更多細節，請查看What ChatGPT O1 Can Do That 4o-mini Cannot。

此時，我並沒有使用像 Cursor 或 Windsurf 這樣的編輯器，儘管我在另一個項目中嘗試過它們。這些代碼編輯器的問題在於它們只能捕獲本地測試輸出，這限制了它們在雲環境中的功能。

在像 GitHub Actions、Jenkins 作業或任何代碼部署或測試流程中，代碼編輯器需要更好地集成。它們應該提供與雲和 CI/CD 流程的無縫交互。

這種集成也適用於其他內容創作工具——無論是文本、圖像、音頻還是視頻。這些工具應該與 A/B 測試系統集成。AI 工具可以生成內容，而 A/B 測試工具可以提供反饋。這種動態類似於從人類反饋中進行強化學習（RLHF），其中 AI 模型根據現實世界的反饋不斷改進。

這種將 RLHF 擴展到不僅僅是模型輸出——進入現實世界的測試和部署環境——的概念，似乎是代碼編輯器和 AI 驅動的內容創作工具改進的一個有前途的方向。

測試可以是即時的或長期的，可以是自動化的或由人類輔助的。如果測試是自動化的，例如對 AI 工具進行用戶 A/B 測試，它仍然涉及人類反饋，但過程是自動化的。例如，我們可以讓計算機根據 A/B 測試結果每天或每小時檢查結果，以改進創作過程。同樣，對於 Jenkins 或 GitHub Actions 作業，我們可以在它們的任務完成後讓計算機檢查。

如果涉及人類輔助，反饋無法完全被機器理解，並且通常有些模糊。例如，當 AI 工具創建圖像或視頻等內容時，人類可能會指出內容不夠有趣，或者應該改進某個細節。機器在使一切完美方面還有很長的路要走，而 “完美” 往往是主觀的，取決於個人的品味。是人類的反饋幫助使事情變得更好。

理論上，所有人類定義的規則都可以寫成提示。有用戶提示和系統提示。我們應該專注於改進提示，而不是每次都修復輸出。