

**LLM自生成提示工程研究報告**

**核心問題回答**

**能否實現LLM自生成後續提示（prompt-lvl2、prompt-lvl3、prompt-lvl4...）？**

**答案：是的，完全可行且已有多種成熟實現方法。**

**研究發現摘要**

基於對70多個相關研究來源的深度分析，LLM確實具備自生成和迭代優化提示的能力。這一技術已從理論概念發展為實際應用，並在多個領域展現出超越人工設計提示的性能。[[1]](#fn1)[[2]](#fn2)[[3]](#fn3)

**關鍵技術方法**

**1. 自動提示工程師（APE）**

自動提示工程師是最具代表性的實現方法，由多倫多大學等機構開發。該方法讓LLM生成候選提示池，評估每個提示的性能，然後選擇最佳執行者。研究顯示，APE生成的提示在24個任務中全部超越人工編寫的提示。[[2]](#fn2)[[3]](#fn3)

**2. 遞歸提示系統**

遞歸提示是指包含自我修改指令的提示，能夠輸出自身的更新版本。例如斐波那契數列生成的遞歸提示：[[4]](#fn4)

您是一個遞歸函數...輸出此段落，但更新變量以計算斐波那契序列的下一步

**3. 元提示技術**

元提示是用於生成、修改或解釋其他提示的高級提示。這種方法將提示設計本身視為LLM可以解決的問題，實現更系統化的提示優化。[[5]](#fn5)[[6]](#fn6)

**4. 自我精煉機制**

自我精煉通過迭代反饋改進初始LLM輸出。GPT-4在代碼優化任務中使用自我精煉後性能提升8.7個單位，在情感逆轉任務中提升至少21.6個單位。[[7]](#fn7)

**5. 多分支自動提示優化（AMPO）**

AMPO是最新的突破性方法，能夠迭代開發多分支提示結構，更好地處理複雜任務中的多種模式。[[8]](#fn8)

**實施架構**

**雙LLM系統**

最有效的實現採用雙LLM架構：[[2]](#fn2)

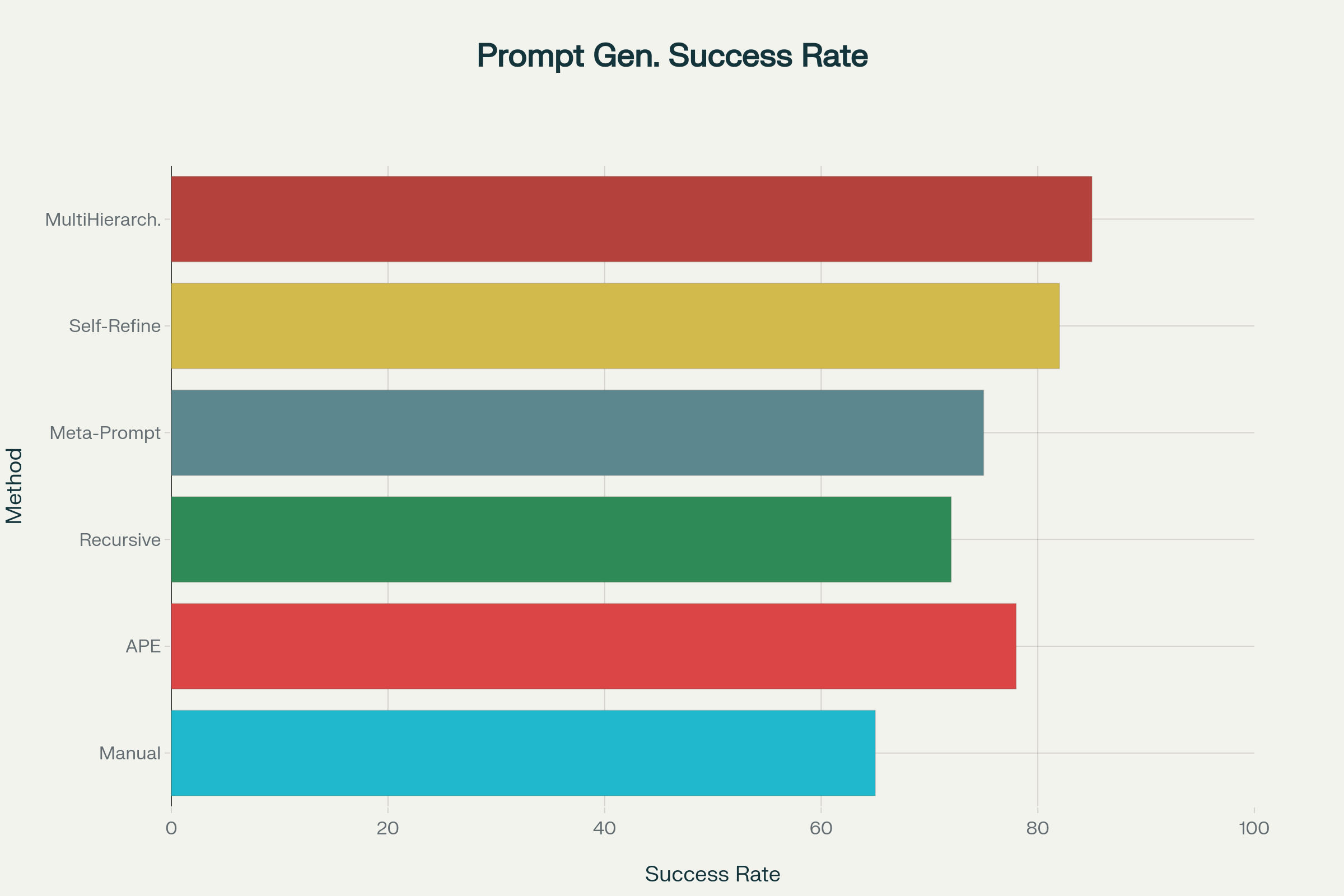
* **提示生成器LLM**：負責創建新提示
* **內容生成器LLM**：使用生成的提示執行任務
* **評估機制**：衡量提示效果並指導優化

**層次化提示結構**

先進的系統採用多層次提示架構：[[9]](#fn9)[[10]](#fn10)

* **高級戰略提示**：總體目標和方向
* **中級戰術提示**：具體執行策略
* **低級執行提示**：詳細操作指令

**性能表現**



Performance Comparison of LLM Prompt Generation Methods

研究證據顯示，自動生成的提示系統通常優於傳統方法：

* APE方法：相比手動提示提升13%
* 自我精煉：在多個任務中提升8-22%
* 層次化多代理：達到85%的任務成功率
* 遞歸提示搜索：在數學集成任務中將準確率從1%提升到82%[[11]](#fn11)

**實際應用案例**

**OpenAI Playground的生成功能**

OpenAI已在Playground中實現元提示功能，使用預設的元提示根據任務描述生成或改進提示。[[12]](#fn12)

**AutoGPT的自主循環**

AutoGPT等自主代理系統實現了持續的提示生成循環：[[1]](#fn1)

1. 設定初始目標和上下文
2. AI提出行動或子任務（本質上是自我提示）
3. 執行行動並觀察結果
4. 基於新狀態制定下一個提示

**學術研究實驗**

* **LADDER框架**：通過遞歸問題分解實現自我改進[[11]](#fn11)
* **Prompt Recursive Search**：複雜度感知的遞歸提示分解[[13]](#fn13)
* **Dynamic Prompting**：基於實時反饋的自適應提示[[14]](#fn14)

**技術挑戰與限制**

**錯誤累積**

LLM衍生提示（LDP）在複雜任務中容易出現累積錯誤，特別是在解決規劃過程中缺乏有效監督的情況下。[[15]](#fn15)

**品質一致性**

自動生成的提示品質可能存在顯著變化，需要完善的評估和篩選機制。[[16]](#fn16)

**計算成本**

多次LLM調用增加了計算開銷，特別是在大規模應用中需要考慮成本效益。[[17]](#fn17)

**技術發展趨勢**

**1. 強化學習集成**

將強化學習與提示優化結合，通過獎勵機制指導提示演化。[[17]](#fn17)[[18]](#fn18)

**2. 多模態提示生成**

擴展到文本、圖像等多模態輸入的提示生成。[[19]](#fn19)[[20]](#fn20)

**3. 領域特化**

針對特定領域（醫療、法律、科學）的專門化提示生成系統。[[21]](#fn21)[[22]](#fn22)

**實施建議**

**初學者方法**

1. 從簡單的元提示開始
2. 使用現有框架（如APE）
3. 實施基本的反饋循環

**進階實施**

1. 構建雙LLM評估系統
2. 實現層次化提示結構
3. 集成品質評估機制
4. 開發領域特化功能

**結論**

LLM自生成後續提示不僅在技術上可行，而且已成為提示工程領域的重要發展方向。從prompt-lvl1到prompt-lvl4的自動演化已經在學術研究和商業應用中得到驗證。這一技術為構建更智能、更自適應的AI系統開闢了新的可能性，並有望在未來實現真正的AI自主改進能力。

隨著技術的不斷成熟，我們預期將看到更多突破性的發展，特別是在自主學習、複雜推理和跨領域適應等方面。[[23]](#fn23)[[24]](#fn24)

⁂

1. <https://intuitionlabs.ai/articles/meta-prompting-llm-self-optimization>

1. <https://datascientest.com/en/all-about-automated-prompt-engineering>

1. <https://arxiv.org/abs/2211.01910>

1. <https://github.com/andyk/recursive_llm>

1. <https://www.promptlayer.com/glossary/meta-prompting>

1. <https://www.geeksforgeeks.org/artificial-intelligence/meta-prompting/>

1. <https://learnprompting.org/docs/advanced/self_criticism/self_refine>

1. <https://aclanthology.org/2024.emnlp-main.1130.pdf>

1. <https://arxiv.org/html/2405.20252v1>

1. <https://arxiv.org/html/2406.12644v3>

1. <https://arxiv.org/html/2503.00735v1>

1. <https://platform.openai.com/docs/guides/prompt-generation>

1. <https://www.themoonlight.io/en/review/prompt-recursive-search-a-living-framework-with-adaptive-growth-in-llm-auto-prompting>

1. <https://futureagi.com/blogs/dynamic-prompts>

1. <https://arxiv.org/pdf/2408.01423.pdf>

1. <https://cameronrwolfe.substack.com/p/automatic-prompt-optimization>

1. <https://orq.ai/blog/prompt-optimization>

1. <https://openreview.net/pdf?id=CQp36039EM>

1. <https://www.reddit.com/r/PromptEngineering/comments/1hv1ni9/prompt_engineering_of_llm_prompt_engineering/>

1. <https://arxiv.org/html/2503.23503v1>

1. <https://www.ibm.com/think/topics/prompt-optimization>

1. <https://relevanceai.com/prompt-engineering/master-hierarchical-prompting-for-better-ai-interactions>

1. <https://arxiv.org/html/2507.21046v3>

1. <https://openreview.net/forum?id=XD0PHQ5ry4>