GII 研究｜實體人工智慧（Physical AI）與商業化｜第二集

推論經濟（The Inference Economy）

《餐廳作為實體人工智慧平台（The Restaurant as a Physical AI Platform）》續篇｜SGLang Summit 2026

當可靠的系統能將模型能力轉化為可衡量的成果，推論（Inference）才具備商業價值。

GII 獨立研究｜人工智慧基礎設施（AI Infrastructure）與營運經濟學（Operating Economics）

2026年11月｜獨立完整版 2.0｜現場紀錄：2026年11月12至13日

李志傑 博士（Dr. Cervantes Lee）｜全球創新研究院（Global Innovation Institute）｜clee@giinow.org

餐旅商業教授｜不動產直接投資人｜人工智慧基礎設施研究

提供科技主管、餐飲與零售業經營者，以及人工智慧基礎設施讀者的原創現場報導與分析。

核心論點

單一回答的串流輸出速度（Streaming Speed）提高至四倍，整體系統的吞吐量（Throughput）卻可能不變。餐廳行銷活動吸引更多購買，邊際貢獻（Contribution）卻可能下降。兩者的關鍵都是：究竟改善了什麼？改善是否傳遞到企業重視的成果？本系列第二集循著這個問題，從 SGLang Summit 的演講與展場，走向顧客資料庫、模型服務引擎（Serving Engine）與資料中心機架（Data-center Rack）。

我根據現場照片、場次紀錄與相關公開文件，說明推論如何成為可運作的服務。連結技術與實務的是一家假設擁有30萬名顧客、10萬份可識別顧客資料的餐飲企業：它如何理解需求、估算顧客價值，並改善接觸對象的選擇？

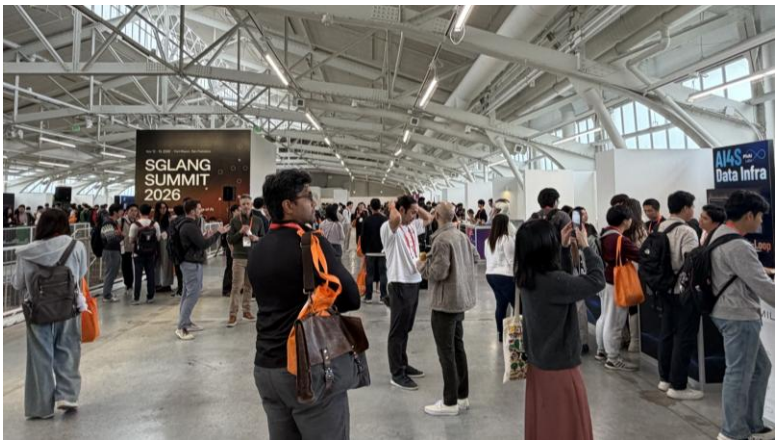


圖1｜Fort Mason 的 SGLang Summit 展場。作者現場攝影。

閱讀指南——從會議現場到顧客決策

執行摘要提出核心論點。第一至二章銜接第一集，並說明產業參與的原因；第三至六章追蹤模型服務、硬體、應用與學習；第七章檢視經濟效益；第八章展開餐飲顧客智慧（Customer Intelligence）案例；第九章討論對營運、機器人與資料中心的意涵。術語表與文末註釋讓讀者可以獨立閱讀本書。

表1｜研究章節與建議閱讀路徑

閱讀目的	建議起點	支持的決策
核心論點	執行摘要；第一至二章	推論為何影響企業營運
模型服務的運作機制	第三章	改善的是哪個時間指標、工作負載與限制？
硬體與記憶體（Memory）	第四章	什麼配置能提供所需服務？
應用與學習	第五至六章	什麼才算有用且可驗收的結果？
商業經濟效益	第七章	誰付費？哪些成本與風險仍存在？
顧客智慧（Customer Intelligence）	第八章	顧客資料如何轉為可衡量的決策？
跨產業意涵	第九章	什麼證據足以支持下一次部署（Deployment）？
定義與證據	附錄A至B；文末註釋	如何理解術語、主張與公開參考資料

第三章與第八章合讀，可連結推論工程與顧客決策。關注硬體的讀者可從第四章開始；營運及商業讀者可從第七至九章開始。

表2｜如何解讀報告中的證據

證據類型	本報告的處理方式
現場照片（Field Photograph）	記錄現場可見事物；不能據此證明部署規模或財務成功。
講者或廠商主張（Speaker or Vendor Claim）	標示其演講來源，且不超出原有適用範圍。
GII 分析解讀（GII Interpretation）	指出商業意涵，以及驗證該解讀所需的證據。
示範性計算（Illustrative Calculation）	明列假設；不代表實測基準（Benchmark）或公司預測。
公開背景文件	以公司原始揭露與技術文件，支持具有時間基準的背景事實。

目錄

執行摘要——推論經濟（The Inference Economy） **8**
技術進步的意義，在於能否改變營運成果 8
作者與現場採訪紀錄 **9**
SGLang Summit 現場 9
1 從餐廳作為實體人工智慧平台到推論經濟 **10**
1.1 延續營運問題 10
1.2 展場讓哪些事情具體可見 10
1.3 決策連結模型與企業 11
2 產業為何圍繞推論集結 **12**
2.1 共用引擎支援不同商業模式 12
2.2 英特爾、AMD與輝達參與的營運邏輯 12
2.3 抹茶、長隊伍與開發者注意力 13
2.4 英特爾——軟體承諾讓主管參與具有實質意義 13
2.5 AMD——開發者是部署路徑的一部分 14
2.6 Hugging Face——優化必須真正進入應用 15
2.7 IBM Spyre——整合讓新硬體真正可用 16
3 從提問到回答之間發生什麼事 **17**
3.1 推論的意義 17
3.2 預填充（Prefill）——處理輸入 17
3.3 解碼（Decode）——產生輸出 18
3.4 Horace He——先問百分比衡量的是什麼 19
3.5 四倍快仍可能維持相同吞吐量 20
3.6 排程效益的條件式公式 21
3.7 預填充與解碼分離必須付出的成本 21
3.8 推論工作負載成長的四種方式 22
3.9 Baseten——衡量整套優化 23
3.10 Traction Layer——可預期的服務本身就是產品 24
4 為何必須看完整系統 **25**
4.1 記憶體容量與資料移動塑造服務 25
4.2 騰訊TACO——管理整個模型服務系統的狀態 25
4.3 Netpreme——延伸記憶體階層 26
4.4 WiCi InferSSD——以頻寬取捨換取容量 27
4.5 輝達——從基礎設施到晶片解讀機架 28
4.6 AMD——從加速器板卡到完整伺服器 31
4.7 Uni³Sim——採購前先測試系統設計 35
5 從快速詞元到有用工作 **36**
5.1 衡量完整工作流程（Workflow） 36
5.2 Arcee——代理仍有工作要完成 36
5.3 Palo Alto Networks——太晚的安全答案可能失去作用 37
5.4 Phala——隱私必須貫穿執行路徑 38
5.5 Core Automation——讓研究實驗更容易執行 39
6 能搜尋、學習與驗證的推論 **40**
6.1 SGLang與Miles——模型服務支援學習循環 40
6.2 Miles——學習循環需要營運紀錄 41
6.3 異質強化學習（Heterogeneous RL）——協調裝置與策略 41

6.4 ApprenticeBench——評估持續進行的工作 43
6.5 Axiom Math——將生成與形式驗證結合 44
6.6 Axiom募資與商業驗證 45
6.7 Apodex——調查共識背後的證據 46
6.8 Krea——獎勵必須保留預期圖像 47
6.9 MiniMax——協調工具擴大輸出形式 47
7 經濟效益在哪裡成立 49
7.1 瓶頸會移動，利潤也會移動 49
7.2 SF Compute——價值獲取取決於情境 49
7.3 Hyperbolic——讓承諾容量對應有用工作 50
7.4 SkyPilot——協調決定有效利用 51
7.5 容量、利用率與邊際貢獻是不同指標 51
7.6 讓營收假設清楚可見 52
7.7 信任與可靠性屬於商業服務內容 52
8 在30萬名顧客的企業中理解10萬份顧客資料 53
8.1 核對顧客母體（Customer Population） 53
8.2 建立顧客證據表（Customer Evidence Table） 53
8.3 選演算法前先理解需求 54
8.4 以明確期間估算顧客價值 54
8.5 從過去學習，避免未來資料洩漏 55
8.6 語言模型推論在哪裡增加價值 55
8.7 接觸顧客時，別混淆回應與影響 56
8.8 計算行銷活動經濟效益 56
8.9 讓模型服務設計對應決策 57
8.10 九十天研究流程 57
8.11 完整顧客智慧循環（Customer-intelligence Loop） 58
9 對餐飲、機器人與資料中心的意涵 59
9.1 餐飲經營者需要改變什麼 59
9.2 Generalist——推論與實體動作交會 59
9.3 資料中心容量必須轉為明確服務 60
9.4 我會帶進下一次部署的問題 60
9.5 下一步研究議程 60
附錄A 實用術語表（Working Glossary） 61
實用術語表 61
實用術語表（續） 61
附錄B 採訪方法與證據 63
編輯與出版紀錄 63
文末註釋與公開參考資料（Endnotes and Public References） 64

表目錄

表1｜研究章節與建議閱讀路徑 2
表2｜如何解讀報告中的證據 2
表3｜營運技術堆疊（Operating Stack） 12
表4｜四種問題與四種方法 17
表5｜速度指標術語 18
表6｜75 / 25案例的理想化配置 19
表7｜相同輸出，不同使用者體驗 21
表8｜對原始預填充時間占比的敏感度 21
表9｜PD部署決策紀錄 22
表10｜四個成長軸向 22
表11｜端到端任務示範 36
表12｜Axiom已揭露募資輪次 45
表13｜假設每月總營收：100個加速器×720小時 52
表14｜顧客母體核對 53
表15｜最小證據資料結構（Evidence Schema） 54
表16｜12個月邊際貢獻示例 55
表17｜實用模型開發流程 55
表18｜行銷活動對照示例 56
表19｜邊際貢獻推導示例 57
表20｜工作負載與架構對照 57
表21｜術語表A——模型服務與硬體 61
表22｜術語表B——學習、信任與顧客分析 61
表23｜證據區分 63

圖目錄

圖1｜Fort Mason 的 SGLang Summit 展場 1
圖2｜李志傑 博士（Dr. Cervantes Lee）於峰會贊助商背板前 9
圖3｜參與識別證——EXPO PASS，QR碼已遮蔽 9
圖4｜會議舞台周邊的交流 11
圖5｜Nebius / Token Factory 抹茶櫃檯 13
圖6｜現場轉播螢幕中的陳立武（Lip-Bu Tan） 13
圖7｜觀看英特爾主管對談的觀眾 14
圖8｜Vamsi Boppana，峰會官方肖像 14
圖9｜Adarsh Shirawalmath 解說運算核心（Kernels） 15
圖10｜運算核心的可攜性（Portability）與分發（Distribution） 15
圖11｜Antoni Viros i Martin 解說IBM Spyre 16
圖12｜IBM Spyre軟體整合 16
圖13｜Horace He解說推論排程（Inference Scheduling） 19
圖14｜四個引擎的預填充與解碼配置 20
圖15｜沿著引擎時間軸進一步展開的兩種模型示意 20
圖16｜Tri Dao與Alexander Korte介紹Baseten 23
圖17｜Baseten優化堆疊（Optimization Stack） 23
圖18｜Ghirardelli Square周邊活動的講者 24
圖19｜上游發布頻率與企業核准流程 24
圖20｜Xibo Sun介紹騰訊TACO 25
圖21｜騰訊TACO資源管理 25
圖22｜Nikita Lazarev解說記憶體分層（Memory Tiering） 26
圖23｜超越HBM的記憶體分層 26
圖24｜Zili Meng介紹WiCi InferSSD 27
圖25｜SSD容量及其頻寬落差 27
圖26｜輝達機架另一側 28
圖27｜正面各區執行不同工作 28
圖28｜附識別牌的GB300運算托盤 29
圖29｜GB300電線、液冷接頭與風扇 29
圖30｜Rubin運算托盤及覆蓋的處理區 30
圖31｜Rubin交換器托盤的管理與冷卻區 30
圖32｜八個處理器位置編號的Vera CPU托盤 31
圖33｜Vera CPU封裝與周邊記憶體陣列 31
圖34｜共用底板上的八個AMD加速器模組 32
圖35｜MI350X規格牌確認產品系列 32
圖36｜銀色伺服器開蓋內部 33
圖37｜M.2儲存裝置及兩個尚未識別的接線模組 33
圖38｜主板接線、記憶體卡扣與局部冷卻 34
圖39｜AMD加速器上方散熱組件 34
圖40｜Zhenhua Zhu介紹Uni³Sim 35
圖41｜Uni³Sim系統配置 35
圖42｜講台上的Lucas Atkins 36
圖43｜快速詞元，緩慢代理（Fast Tokens, Slow Agents） 37
圖44｜為何使用語言模型做安全推論 37
圖45｜Hang Yin介紹機密推論（Confidential Inference） 38

圖46｜機密執行與驗證證明（Attestation） 38
圖47｜Francis Zhang解說研究基礎設施 39
圖48｜推論與研究 39
圖49｜Ying Sheng，峰會官方肖像 40
圖50｜模型服務與強化學習回饋 40
圖51｜標示Zhichen Zeng姓名的Miles場次標題 41
圖52｜Miles後訓練環境 41
圖53｜「在一種晶片訓練，在任何晶片生成軌跡」場次標題 41
圖54｜把每個推論伺服器當成軌跡生成伺服器 42
圖55｜標示Xiang Deng的ApprenticeBench標題 43
圖56｜ApprenticeBench在工作中學習 43
圖57｜Tobias Gessler與Rishi Malhotra介紹Axiom Math 44
圖58｜Axiom Math訓練與證明基礎設施 44
圖59｜生成、形式驗證（Formal Verification）與回饋 45
圖60｜Simon Shaolei Du介紹Apodex 46
圖61｜把推理視為團隊運動 46
圖62｜Sangwu Lee介紹Krea 47
圖63｜Krea強化學習更新 47
圖64｜Open Stage上的Ryan Lee 47
圖65｜模態與表達頻寬（Bandwidth of Expression） 48
圖66｜Ethan Anderson解說價值獲取（Value Capture） 49
圖67｜利潤會留在哪裡——價值獲取矩陣 49
圖68｜Jasper Yue Zhang解說運算網格（Compute Grid） 50
圖69｜運算網格架構 50
圖70｜Zhanghao Wu介紹SkyPilot 51
圖71｜SkyPilot中的模擬控制負載 51
圖72｜餐飲顧客智慧營運循環 58
圖73｜Generalist GEN-1.5場次標題 59
圖74｜實體提示與已學習模型權重 59

執行摘要——推論經濟（The Inference Economy）

技術進步的意義，在於能否改變營運成果

四倍的改善，可能不改變主要商業指標。在 Horace He 簡化的預填充 / 解碼（Prefill/Decode）案例中，每位使用者的回答串流速度變成四倍，但總輸出仍為每秒160個詞元（Tokens）。在我的餐廳行銷案例中，較高購買率在扣除折扣與溝通成本後，反而產生較低邊際貢獻（Contribution）。兩者都符合內在邏輯。重點是辨認改善了哪項限制，並沿著完整工作流程（Workflow）追蹤其影響。[1]

這是 GII「實體人工智慧與商業化」研究系列第二集。第一集《餐廳作為實體人工智慧平台（The Restaurant as a Physical AI Platform）》探討人工智慧能力如何在真實企業取得實用地位。在 SGLang Summit 2026，我追問下一個問題：從訓練完成的模型，到提供顧客、員工或機器的可靠服務，中間必須發生哪些事？推論（Inference）是反覆應用模型能力的過程，也讓等待時間、成本與失敗成為具體的營運問題。[2]

峰會的技術討論指向更廣泛的商業競爭。硬體供應商需要讓設備可用的軟體；模型服務平台需要將專業工程轉化為可重複提供的服務；應用開發者需要在時限與預算內取得有用結果。開放原始碼基礎設施（Open-source Infrastructure）能同時支援三者，也為整合、支援與執行品質留下充分競爭空間。

預填充（Prefill）與解碼（Decode）是這場競爭的縮影。處理提示詞（Prompt）與產生回答，對系統提出不同需求。將兩階段分離可能改善使用者體驗，但效果取決於工作負載（Workload）、記憶體、資料傳輸與排程（Scheduling）。因此，「推論更快」需要清楚的衡量分母：更快的首次回應、更快的串流、更多完成的工作，還是在指定品質下成本更低？

餐飲應用讓這項區別更具體。一家假設擁有30萬名顧客、10萬份可識別資料庫檔案的企業，需要可靠紀錄、預測模型（Predictive Models）、選擇性使用語言模型分析，以及對照實驗（Controlled Experiments）。目標是理解需求並改善顧客決策。本報告未取得或分析任何顧客資料集；示範案例說明經營者如何設計研究。支持論點的產業案例與解說均納入本書，場次歸屬與公開來源註釋列於文末。

作者與現場採訪紀錄

李志傑 博士（Dr. Cervantes Lee） 全球創新研究院（Global Innovation Institute）

我從《餐廳作為實體人工智慧平台》提出的營運問題切入推論：令人印象深刻的能力，何時能成為企業中可靠且具有經濟價值的一環？我在餐飲、教育、不動產與投資的經驗，使這個問題始終與交付成果所需的人員、建築及系統密不可分。

在 Fort Mason，我循著這個問題觀察演講、示範與設備展示。本書把這些觀察整合為推論經濟的完整敘事。照片交代採訪情境；技術解說與計算則讓內容在會議之外仍具實用價值。



圖2｜李志傑 博士（Dr. Cervantes Lee）於峰會贊助商背板前。作者現場照片。

SGLang Summit 現場



圖3｜參與識別證——EXPO PASS，QR碼已遮蔽。使用人工智慧協助旋轉及遮蔽QR碼，原始證件照片已保留。

識別證記載作者姓名、全球創新研究院（Global Innovation Institute）、SGLang Summit 2026、11月12至13日，以及舊金山 Fort Mason。此處以遮蔽QR憑證的形式，作為參與紀錄。

採訪、攝影與分析：全球創新研究院李志傑 博士（Dr. Cervantes Lee）。人工智慧工具協助資料整理、撰稿與文件製作。場次陳述及公司揭露均保留來源歸屬；分析判斷與示範計算標明為作者工作。文末註釋說明各主題的證據，附錄B交代採訪方法。

1 從餐廳作為實體人工智慧平台到推論經濟

1.1 延續營運問題

我的起點很實際：當人工智慧推論變得更快、更可靠、更便宜，經營者能做到什麼？我的研究連結餐飲、零售、實體人工智慧與基礎設施。企業將觀察轉化為決策，並以設備、軟體、人員與資本支撐決策時，這些領域便交會。

第一集《餐廳作為實體人工智慧平台》建立三個商業化單位：據點是經濟效益的單位（Site as the Unit of Economics），任務是學習的單位（Task as the Unit of Learning），網路是規模化的單位（Network as the Unit of Scale）。第二集將此架構延伸至推論：據點承擔服務成本與營運限制；任務界定模型必須完成的工作；網路決定有用結果能否跨據點、跨顧客重現。[2]

餐飲主管不必以詞元（Tokens）管理企業。他需要知道：分析是否改善留存率（Retention）、能否透過已授權管道接觸顧客、優惠扣除折扣後是否增加邊際貢獻，以及系統能否在決策期限前完成工作。當詞元、批次（Batches）與GPU記憶體能解釋成果的成本或可行性時，這些技術概念才與經營決策相關。

這也解釋基礎設施會議為何與具不動產及資本配置背景的研究者相關。機架（Rack）是對特定營運能力的投資，受電力、冷卻、網路與軟體限制。設備裝得更多，只有在能完成可出售或對內具有價值的工作時才有意義。本報告探討從硬體通往邊際貢獻的路徑。

我的解讀是：推論連結資訊與重複性的決策或行動，因此屬於營運研究的一部分。不同演講從不同方向處理這項連結。彼此的分歧與取捨很有價值，因為它們說明單一晶片規格或模型分數，無法解釋整個商業體系。

1.2 展場讓哪些事情具體可見

Fort Mason 內，人群聚集在設備、示範螢幕與交流區周圍。展場讓抽象議題變得具體。詞元討論背後，是努力讓服務運作的工程師、爭取工程師注意的廠商，以及藉由管線與纜線呈現實體需求的機架。

照片引導讀者理解現場情境。講者與演講標題共同確認討論主題；投影片照片呈現所說明的主張或機制；硬體特寫讓我們區分運算托盤（Compute Tray）與交換器（Switch），並追蹤支援兩者的連接。

視覺紀錄也有界限。模糊標籤無法證明產品世代，熱鬧會場無法證明市占率。當識別很重要時，我將照片與相關展示標籤或公開技術文件對照，保留尚未解決的不確定性。

現場紀錄是本報告的起點。以下章節探討這些觀察對購買、建置與營運人工智慧的企業有何意義。

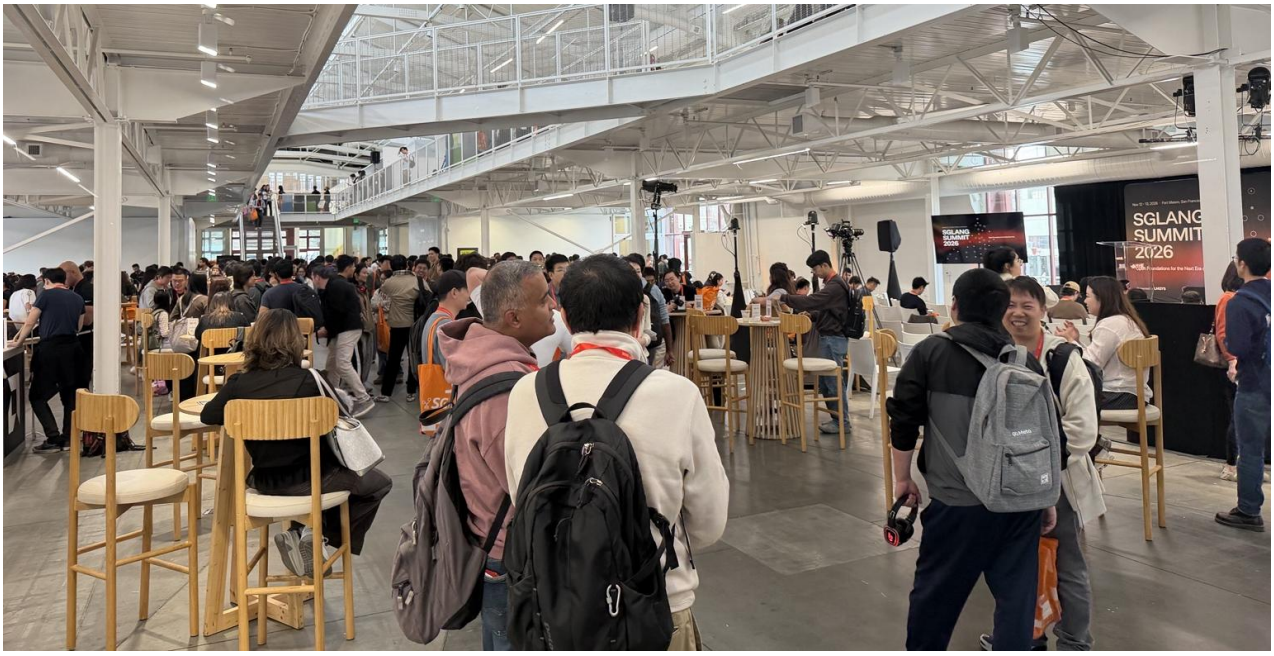


圖4｜會議舞台周邊的交流。作者現場攝影。

1.3 決策連結模型與企業

本報告追蹤三個相連問題：系統必須完成什麼？什麼阻礙它可靠且符合經濟效益地完成？哪些證據能證明限制已改善？各章以這些問題組織公司案例，而非把每場演講當成彼此獨立的終點。

對餐廳而言，成果可能是更有用的顧客決策；對研究團隊，是通過檢查的證明或提早完成的實驗；對基礎設施買方，是在電力、延遲與成本範圍內交付明確服務。無論哪一種，任務及驗收測試（Acceptance Test）都應先於吸引人的效能標題。

2 產業為何圍繞推論集結

產業圍繞共用模型服務引擎集結，引出商業問題：當推論更容易部署，各參與者獲得什麼？晶片供應商、開源維護者、雲端業者與應用開發者的答案不同。共同利益是建立從模型到服務的可行路徑；營收機會則分布在路徑的不同環節。

2.1 共用引擎支援不同商業模式

模型（Model）包含學習得到的數值參數；應用（Application）提供指令、輸入與目的；模型服務系統（Serving System）安排回答請求所需的計算；學習系統（Learning System）之後可利用資料與回饋更新模型。這些層彼此相連，但工作並不相同。

SGLang 處理模型服務；Miles 組織反覆使用服務能力的強化學習（Reinforcement Learning）工作。模型產生嘗試，環境或驗證器（Verifier）評估嘗試，訓練再改變模型。正式上線仍需獨立的評估決策。當回饋衡量的是值得學習的行為，這個循環才有價值。[3]

這有助於解釋峰會公司的多樣性。硬體商需要有用軟體支援處理器；模型公司需要從權重（Weights）通往應用的可靠路徑；雲端平台需要能排程並出售的工作負載；應用開發者需要各層協同運作，而不必讓每位顧客都成為系統工程師。

共用軟體可同時擴大多家競爭供應商的市場，降低部分採用門檻，並將競爭移向部署品質、可用性、支援、整合及特定負載的效能。開放協作與商業競爭可以並存。

SGLang 聯盟（SGLang Consortium）提供另一種協調機制：參與者改善共同技術基礎，維護者仍保有技術採納決定權。贊助商標誌代表參與，不代表擁有程式碼或控制專案。圍繞共用引擎的商業利益仍可各自獨立。[3]

表3｜營運技術堆疊（Operating Stack）

層級	主要工作	商業問題
應用（Application）	選擇有用的任務	誰從答案受益？
模型（Model）	將輸入映射為輸出	能力是否足夠？
模型服務（Serving）	安排執行與重用	能否符合時限？
基礎設施（Infrastructure）	提供運算與資料移動	能否可靠交付有用工作？
評估（Evaluation）	測試輸出及成果	流程是否改善？

2.2 英特爾、AMD與輝達參與的營運邏輯

硬體高階主管與相互競爭的晶片商為何聚集在推論軟體周圍？經濟上的答案很直接：硬體需求取決於顧客能成功執行什麼。處理器透過模型、運算核心（Kernels）、編譯器（Compilers）、服務引擎、網路與支援安排，才轉化為可部署、具商業價值的設備。

我拍下現場轉播螢幕中的陳立武（Lip-Bu Tan），以及觀看對談的觀眾。英特爾（Intel）的公開公告將這次參與連結到與 RadixArk 合作優化 SGLang，以及加入 SGLang 聯盟。這些承諾說明其出席的意義；我並無陳立武完整談話逐字稿，無法據此重建其全部論述。[4]

AMD 的開發者計畫展示凸顯採用問題：誰來在硬體上建置、測試與維護應用？其參與結合主管曝光、工程活動與開發者推廣。第四章的設備照片說明，加速器板卡（Accelerator Board）、CPU伺服器與完整服務，是採購的不同層級。[5, 6]

在輝達（NVIDIA）展區，系統方案以實物呈現。運算托盤、交換器、電力組件、冷卻接頭與機架結構各有工作。我的解讀是，商業產品延伸至系統提供的有用服務。交換器托盤與GPU運算托盤外觀可能相似，功能卻截然不同。[7]

三家公司都有理由讓部署更容易。輝達的展示強調整合；AMD的資料連結硬體與開發者採用；英特爾公告連結主管參與與軟體合作。這些觀察解釋參與動機。若要評比替代方案，仍需相同工作負載、實際交付價格與營運資料。

2.3 抹茶、長隊伍與開發者注意力

Nebius 品牌的抹茶櫃檯喚起了我的餐飲研究本能。我很喜歡飲品，雖然長長的隊伍考驗耐心。顯然，即使在人工智慧會議，我仍忍不住觀察顧客體驗。在 Nebius / Token Factory 招牌下，一家新型雲端（Neo-cloud）基礎設施公司，透過熟悉的餐旅服務方式介紹自己。這杯抹茶給了我記住品牌的愉快理由；推論效能與定價則需要另行評估。



圖5｜Nebius / Token Factory 抹茶櫃檯。作者現場攝影。

技術採用也有人性的一面。工程師必須注意到產品、理解介面、取得存取權、測試工作負載，再決定是否依賴它。會議讓這些步驟靠得更近。好喝的飲品可以開啟對話；工程試用必須贏得下一次對話。

對餐飲研究者而言，這一幕提醒我們：注意力、服務與顧客理解跨越產業。第八章將好奇心帶進資料庫：企業如何更了解顧客，並改善接觸誰、提供什麼優惠、付出多少成本的決策？

飲品是親切的開場。當開發者開始試用服務，更深層的商業問題才展開：安裝、衡量、故障恢復與持續運作需要多少工作？以下案例說明，軟體分發（Distribution）與相容性（Compatibility）為何與晶片效能同樣重要。

2.4 英特爾——軟體承諾讓主管參與具有實質意義


圖6｜現場轉播螢幕中的陳立武（Lip-Bu Tan）。作者會議攝影。[4]

現場螢幕轉播照片確認陳立武的可見參與及觀眾情境。我的紀錄沒有完整演講逐字稿，因此英特爾公開合作公告，是討論其工程承諾更穩固的依據。[4]

商業上的解讀是：硬體供應商關心能讓產品部署的軟體生態系（Software Ecosystem）。推論可在不同處理器及系統設計上執行；實際採用取決於目標應用的相容性、效能與支援。

主管參與的意義，在於傳達策略關注，不代表可以代其宣稱未記錄的承諾。嚴謹的後續問題是：英特爾鎖定哪些負載？由哪條軟體路徑支援？在顧客定義的測試下，交付結果如何比較？

照片記錄螢幕轉播，並非音訊逐字稿。螢幕字幕不視為已驗證的逐字引述；引用的議程未標示主持人身分，因此此處不具名。[4]

已核實背景：英特爾為白金級贊助商（Platinum Sponsor）。議程亦包含 Mingfei Ma 關於 Intel GPU與Xeon CPU上 SGLang 的技術場次，將企業曝光連結到軟體工程。[4]

英特爾於11月12日美東下午3時30分、即太平洋時間中午12時30分，宣布與 RadixArk 合作，目標是在英特爾平台優化 SGLang，並向共同專案貢獻。公告早於排定的爐邊對談。[4]

英特爾也是當日宣布成立的 SGLang 聯盟創始成員，其他成員包括 NVIDIA、AMD、Google、Meta、Cloudflare、Modal與 RadixArk。成員提供專業知識與路線圖回饋；維護者保有技術權限。加入聯盟並不讓英特爾控制 SGLang。[4]



圖7｜觀看英特爾主管對談的觀眾。作者會議攝影。[4]

觀眾包含選擇模型服務軟體、影響硬體採購的人。執行長能傳達持續工程支援並吸引夥伴。商業目標是從「模型能執行」走向「顧客能可靠且負擔得起地營運」。沒有來源證明其私人動機，也沒有來源證明與會者作出採購承諾。[4]

陳立武於2025年3月18日起擔任英特爾執行長，之前於2009至2021年任 Cadence 執行長。此背景連結晶片設計軟體與半導體執行能力，但本身不能證明人工智慧模型服務業務會成功。[4]

2.5 AMD——開發者是部署路徑的一部分


圖8｜Vamsi Boppana，峰會官方肖像。峰會官方資料。[5]

AMD 的峰會內容結合贊助參與、高階主管背景與開發者推廣。現場開發者計畫資料指出，將可取得的硬體變為應用平台仍需實際工程工作。

Radeon 工作站照片與 Instinct 資料中心硬體，屬於不同部署情境。蒐集資料中另有 Google TPU 圖像，該圖屬於Google，不作為AMD基準測試。清楚區分產品身分，是有效比較的必要條件。[5]

AMD 的機會取決於實用的模型支援、穩定軟體及交付效能，而不只有晶片規格。競爭問題是顧客能否以具吸引力的總成本部署並維護。此處官方肖像記錄議程中的講者身分，並非作者拍攝的演講照片。

AMD 為白金級贊助商，參與內容包括高階主管、工程演講與強化學習實作工作坊。這反映生態系投資；贊助本身不能證明產品較優。[5]

商業上的解讀是：開發者影響硬體採購。若工程師能在AMD上可靠部署模型，其組織便能考慮正式採用AMD。培訓、雲端存取與軟體支援，可降低評估這項選擇的工作量。[5]

本次採訪紀錄沒有完整AMD場次錄音。此敘事由照片及官方文件支持；有關AMD商業目標的陳述是作者解讀，不是重建的逐字引述。[5]

權重記憶體（Weight Memory）的示範估算，以十進位GB計：參數數量×每參數位元數÷8。80億參數模型以16位元精度約需16 GB，以4位元精度約需4 GB；320億參數模型分別需64 GB與16 GB；700億參數模型分別需140 GB與35 GB。[5]

上述未包含執行時記憶體（Runtime Memory）、量化中繼資料（Quantization Metadata）及鍵值快取（KV Cache）。KV快取保存對話的中間注意力資訊；更長對話與更多同時使用者需要額外空間。4位元壓縮可能影響品質，視模型與方法而定。[5]

Instinct MI355X 是採CDNA 4架構的資料中心加速器，每GPU具288 GB HBM3E與8 TB/s記憶體頻寬（Memory Bandwidth）。此產品類別與R9700不同；九倍記憶體不代表九倍回應速度。[5]

每秒浮點運算次數（FLOPS）衡量運算量；每秒詞元數（Tokens per Second）衡量語言模型輸出速率，兩者沒有通用換算。模型大小、精度、記憶體流量、批次及軟體都影響實際速度。延遲（Latency）衡量等待時間；吞吐量（Throughput）衡量一段時間完成的總工作。[5]

2.6 Hugging Face——優化必須真正進入應用

Adarsh Shirawalmath 的運算核心演講探討：優化程式碼如何可靠地進入應用？運算核心（Kernel）是特定運算的底層實作。實用價值取決於它能否在顧客的硬體與軟體版本下正確建置、載入及使用。



圖9｜Adarsh Shirawalmath 解說運算核心（Kernels）。作者會議攝影。[8]

Kernel Hub及相關工具處理封裝、搜尋與相容性。若優化元件缺少或不相容，應用可能退回較慢的實作。演講中的7.5倍比較採用特定回退基準（Fallback Baseline），不代表所有已優化安裝都再快7.5倍。[8]

降低部署摩擦的商業意義，可能不亞於發明更快的運算；它讓更多開發者取得預期效能。可重現性（Reproducibility）與版本控制仍不可少，尤其核心參與強化學習計算、數值差異可能影響結果時。



圖10｜運算核心的可攜性（Portability）與分發（Distribution）。作者會議攝影。[8]

專案的目錄與工具各異，包括CMake、Nix、Meson及Bazel。編譯可能耗時，支援矩陣亦增加維護量。例如三個PyTorch版本×兩個CUDA版本×兩種處理器架構，在其他限制之外就已有12種組合。必須有人製作、驗證與維護使用者真正需要的組合。[8]

專案提供工具，以可預期的結構建置、分發及使用運算核心。可重現性是充分記錄相依套件與建置條件，以一致方式重建預期產物，減少依賴工程師筆電中未記錄的設定。標準化封裝可讓更多開發者使用專家優化成果。[8]

Kernel-builder 為生產端準備編譯產物；kernels程式庫為使用端尋找並載入適當產物。前者像工廠，後者像配送與安裝。分開責任後，使用者無須重做整個生產流程，也能使用優化程式碼。[8]

二進位檔（Binary）是編譯後的軟體。若底層介面預期改變，針對某版PyTorch建置的元件可能在另一版失效。投影片分別列出PyTorch 2.10、2.11與2.12的二進位檔。維護負擔來自元件相容性，而不是ReLU或注意力（Attention）的數學定義改變。[8]

應用程式二進位介面（Application Binary Interface, ABI）是編譯元件溝通的底層約定。投影片提出以單一二進位檔對應PyTorch穩定ABI，跨支援版本使用，並聲明至少兩年ABI穩定性。如同維持插座規格不變，可保留相容性；但必須使用支援的穩定介面，也不會消除硬體、CUDA或作業系統限制。[8]

範例先嘗試Hub FlashAttention，再嘗試本機安裝實作；若均失敗，標記功能不可用，後續程式仍須改選實作或回報錯誤。回退（Fallback）提高韌性，但默默回退可能隱藏重大效能變化；正式系統應記錄實際執行路徑。[8]

2.7 IBM Spyre——整合讓新硬體真正可用



圖11｜Antoni Viros i Martin 解說IBM Spyre。作者會議攝影。[9]

IBM案例處理晶片規格無法獨自解決的障礙：模型服務框架必須以開發者實際可用的方式，連接硬體執行環境（Runtime）。Antoni Viros i Martin 說明如何將既有Spyre支援接入SGLang，並以代理（Agents）協助工程工作。

「不到一週」指此次整合工作；硬體、編譯器與執行環境並非在該週從零完成。重用既有介面與實作模式，是解釋成果的重要部分，不應從標題中略去。

人工智慧輔助移植（AI-assisted Porting）可能縮短硬體能力到框架支援的路徑。正式使用仍需測試、確認支援的張量形狀（Shapes）及人工審查。Hugging Face的核心基準測試是另一項結果，不歸於IBM。[9, 8]

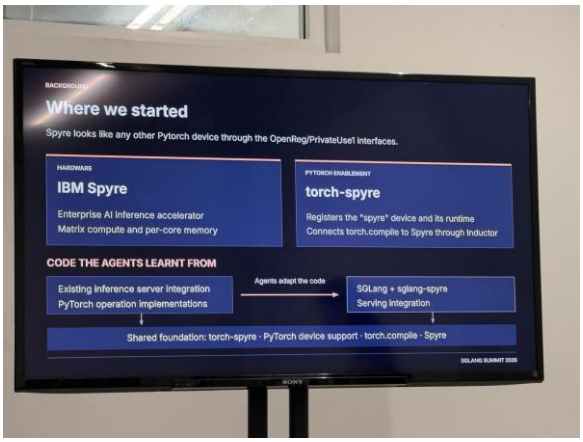


圖12｜IBM Spyre軟體整合。作者會議攝影。[9]

Spyre硬體及torch-spyre原已存在。矩陣引擎（Matrix Engines）處理數值陣列；各核心記憶體讓資料靠近運算位置。[9]

資料必須先抵達，才能開始計算；記憶體配置可能與算術能力同樣重要。[9]

Torch-spyre註冊PyTorch裝置、管理執行時作業，並透過Inductor銜接torch.compile。執行環境處理配置與傳輸；編譯將運算轉成可執行步驟。OpenReg與PrivateUse1是擴充機制，不是新的人工智慧模型。[9]

SGLang管理請求與重用政策；sglang-spyre調整介面；torch-spyre提供張量（Tensors）、記憶體佈局、編譯與執行。[9]

平台註冊辨識裝置；注意力後端（Attention Backend）執行注意力計算；鍵值池（KV Pool）管理已儲存的鍵和值；運算子分派（Operator Dispatch）將作業送到對應實作。[9]

兩名學生可能提交相同文章，卻問不同問題。前綴快取（Prefix Caching）重用相同開頭的計算，不同問題與回答仍須處理。這是重用計算，不是複製現成答案。[9]

硬體執行方式改變時，獨立的模型服務政策仍可維持。張量邏輯形狀（Logical Tensor Shape）與實體記憶體佈局（Physical Memory Layout）是不同概念。[9]

投影片報告：兩位開發者、不到一週、約1,500行外掛程式碼；功能包括注意力、KV快取、模型運算與前綴共享。[9]

縮放點積注意力（Scaled Dot-product Attention, SDPA）比較查詢與鍵，並組合值。既有Spyre注意力與尺寸處理被重用。KV配置保留過去注意力資訊，仍會占用實體記憶體。[9]

模型運算各有角色：均方根正規化（RMSNorm）重新縮放數值；SiLU與GeLU提供非線性轉換；旋轉位置嵌入（Rotary Embeddings）編碼位置；線性層（Linear Layers）乘上已學習權重。這些是不同數學元件。[9]

少量整合程式碼可能仰賴大量既有基礎設施；單靠行數無法衡量開發生產力。[9]

這些案例說明推論會議為何能讓競爭者同聚一室。共通軟體路徑可擴大可用機器與應用範圍。下一個問題是：路徑內如何運作，買方實際獲得哪種改善？

3 從提問到回答之間發生什麼事

「推論」容易讓人以為工作只是一個事件；但語言模型服務是一連串資源需求不同的操作。理解各項操作與衡量時間的方式，就更容易掌握峰會反覆討論的預填充（Prefill）、解碼（Decode）及兩者分離。

3.1 推論的意義

在機器學習（Machine Learning）中，推論（Inference）是把訓練完成的模型套用到新輸入。顧客留存模型可能回傳機率；語言模型可能產生文字序列；機器人策略（Robot Policy）可能輸出動作表示。這些都可稱為推論，但成本、驗證方式與失敗模式不同。

語言模型產生的說明可能看似人類推理（Reasoning），但外觀無法證明每句話正確或推理可靠。商業使用的重點是輸出能否符合明確任務，並以資料或結果查核。流暢的顧客故事，可能不如經校準的機率及清楚決策規則有價值。

訓練（Training）與推論互相影響，但並不相同。訓練改變模型參數；一般推論使用參數。應用可透過增加上下文（Context）改變行為，而不更新參數，正如Generalist的實體提示（Physical Prompting）示範。強化學習亦可使用大量推論嘗試，為之後的參數更新產生回饋。[10, 11]

因此，峰會不只是討論聊天回答如何加速，而是涵蓋持續擴展的運算活動：產生候選結果、處理長上下文、呼叫工具、檢查證明、生成媒體、解讀回饋及協調硬體。同一基礎設施面對的需求可能截然不同。

餐飲案例應先定義決策，再選擇推論方法。估算再次購買機率可從傳統預測模型開始；從文字回饋擷取服務問題可能適合語言模型；歷史邊際貢獻需要準確會計與彙總；檢驗優惠是否帶來額外來客則需要實驗設計。單一方法無法取代其他所有方法。

表4｜四種問題與四種方法

問題	適合的起始方法	驗證方式
發生了什麼？	查詢與描述性分析（Descriptive Analysis）	與原始紀錄核對
可能發生什麼？	預測模型（Predictive Model）	以未來期間資料檢驗校準程度
這段文字代表什麼？	依資料結構（Schema）進行大型語言模型（LLM）擷取	人工標註樣本
我們的行動造成什麼？	隨機實驗（Randomized Experiment）	處理組與對照組的差異

3.2 預填充（Prefill）——處理輸入

預填充處理輸入序列，包括指令、使用者請求、檢索文件與歷史內容。典型的轉換器語言模型（Transformer Language Model）會計算輸入詞元的表示，建立後續注意力可重用的鍵 / 值狀態，通常稱為鍵值快取（KV Cache）。

長提示詞可能遠超使用者可見的問題。研究代理可能納入工具描述、搜尋結果、先前工具輸出與中間討論；下一步重送累積材料時，輸入處理可能成為主要負載。若系統能辨識並安全重用相容的快取狀態，共同前綴可減少重複工作。

預填充常包含能有效利用GPU的大型矩陣運算，尤其輸入量大時。但「常」不是必然：提示長度、模型架構、批次與硬體都會改變瓶頸（Bottleneck）。短請求未必受運算能力限制，快取命中（Cache Hit）也可能大幅改變工作量。

首詞元時間（Time to First Token, TTFT）不只有預填充，還可能包含排隊、路由、輸入準備與其他開銷。即使預填充加速，忙碌工作節點前的排隊仍可能讓使用者等待。可信比較須同時衡量完整首次回應路徑與引擎操作。

顧客智慧應用中，成本最高的提示詞未必最有用。把每筆交易與每份顧客資料塞進同一上下文，通常不是分析整體顧客的有效方法。結構化彙總、預測評分流程及檢索相關支持紀錄，可減少無謂輸入，也更容易驗證。[1, 12, 13]

3.3 解碼（Decode）——產生輸出

解碼利用既有上下文與先前已產生的詞元，繼續產生新詞元。在一般自回歸生成（Autoregressive Generation）中，後續輸出依賴先前輸出，因此包含循序性工作，不能像處理固定輸入那樣一次算完任意完整答案。

小批次下，反覆讀取權重及其他記憶體操作可能限制解碼效率。批次變大或架構不同時，運算與通訊可能更重要。「解碼受記憶體限制（Memory-bound）」在部分情境有助理解，但不是普遍定律。Horace He提醒我們辨認實際運作條件（Operating Point）。[1]

詞元間隔（Inter-token Latency, ITL）是相鄰輸出詞元間的時間，越短則回答看起來越流暢。總輸出詞元 / 秒合計所有請求，屬另一指標。服務可以讓單一回答很快、卻服務較少使用者；也可總吞吐量很高、每位使用者的輸出卻較慢。

KV快取隨序列狀態成長。即使模型參數不變，更長上下文、更多同時序列與架構特有狀態，也會消耗記憶體。因此，可分配給活動請求的記憶體，可能限制批次大小與每個工作節點可容納的使用者數。

推測式解碼（Speculative Decoding）改變候選詞元的提出與驗證方式；獲接受的候選可讓一步前進多個詞元，但效益取決於提案品質、驗證成本與工作負載。不知道每步接受多少詞元，就不能直接將每步毫秒數取倒數，當成使用者每秒詞元數。Baseten演講清楚說明這項區別。[14]

表5｜速度指標術語

指標	意義	常見錯誤
首詞元時間（TTFT）	直到第一個輸出詞元的時間	誤當成純預填充時間
詞元間隔（ITL）	相鄰輸出詞元間的時間	把其倒數當成服務總輸出
吞吐量（Throughput）	每單位時間完成的工作	忽略品質與延遲
有效吞吐量（Goodput）	符合指定目標的有用輸出	未定義目標
端到端時間（End-to-end Time）	完成整項任務的時間	忽略工具、重試與驗證

3.4 Horace He——先問百分比衡量的是什麼

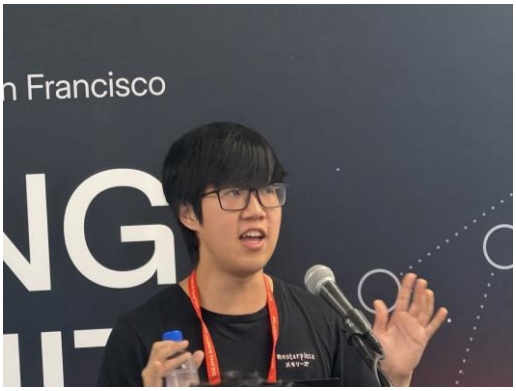


圖13｜Horace He解說推論排程（Inference Scheduling）。作者會議攝影。[1]

Horace He的說明是本報告核心，因為他區分個別串流速度與整體吞吐量。預填充 / 解碼案例顯示，消除中斷可以加速一個請求的解碼串流，卻不增加系統總輸出。

演講說「75%預填充、25%解碼」時，首先要問：什麼的75%？可能是引擎時間、總延遲、硬體配置、成本或詞元數，這些分母不能互換。Horace He的排程案例，是共用引擎花在兩種工作的時間。[1]

假設四個相同混合工作節點，各用四分之三時間預填充、四分之一時間解碼。合計相當於三個工作節點的預填充量，以及一個工作節點的解碼量。分離方案將三個節點專做預填充、一個專做解碼；在理想化計算下，總配置不變。

混合節點上的請求只有部分時間能獲得解碼機會；專用解碼節點可更規律地服務活動序列。若保留相同解碼批次運作條件，移除預填充中斷，可讓單一序列的串流變成四倍快，但不會創造四倍總解碼能力。原本四個節點已提供相當於一個全時解碼節點的能力。

此思想實驗刻意簡化，假設到達流量平衡、批次相容、記憶體足夠，且節點間移動狀態沒有額外成本。真實部署必須逐項驗證。尤其不能把比例解讀為「75%的詞元是輸入」，因為一個輸入詞元與一個輸出詞元未必占用相同引擎時間。

核心區別是：預填充與解碼分離（PD Disaggregation）可藉由改變工作時機，改善單一請求體驗，而不先改變整個服務能完成的工作量。是否也改善成本或容量，需要另行衡量。

表6｜75 / 25案例的理想化配置

設計	預填充等效工作節點數	解碼等效工作節點數
4個混合工作節點	4 × 0.75 = 3	4 × 0.25 = 1
3個預填充＋1個解碼節點	3	1
總配置的變化	無	無

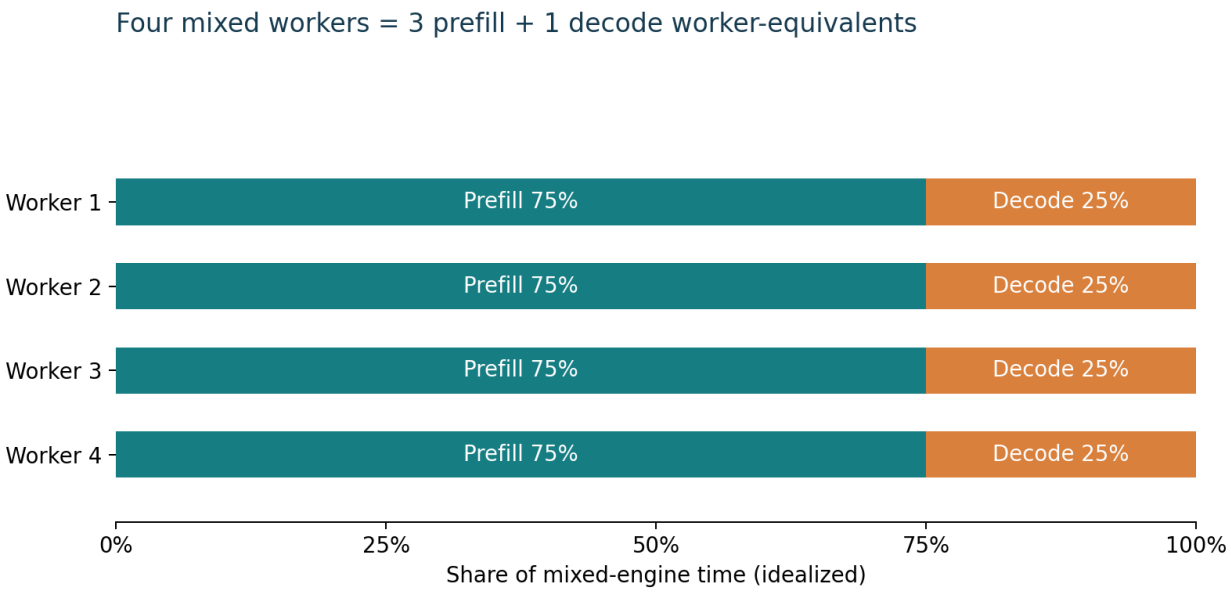


圖14｜四個引擎的預填充與解碼配置。GII解說圖；假設見正文。

每條橫條代表同一時段中的一個混合節點。四分之三時間分配預填充、四分之一分配解碼。四個節點加總，得到三個預填充等效節點與一個解碼等效節點。

階段分離改變排程，並未讓預填充消失。專用解碼器可更連續串流，另三個節點則持續處理輸入。下一節推導到達率、活動請求數及不變的總輸出。

此圖不是「三張預填充GPU搭配一張解碼GPU」的通用配方。「工作節點（Worker）」或「引擎（Engine）」本身可能使用多個裝置。實際配置取決於模型、記憶體、網路與負載；案例只隔離一項排程機制，以便理解後果。

3.5 四倍快仍可能維持相同吞吐量

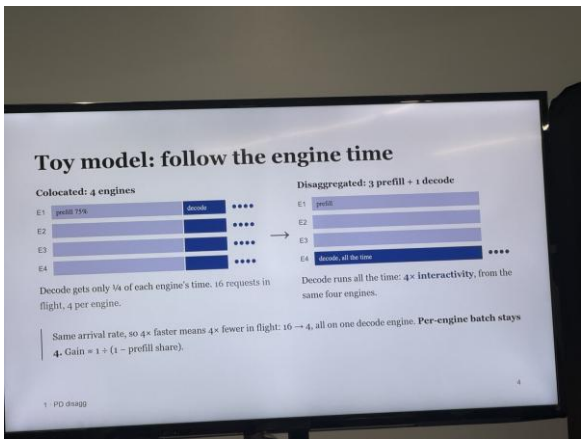


圖15｜沿著引擎時間軸進一步展開的兩種模型示意。作者會議攝影。[1]

以Horace He排程例的示範條件說明：每秒到達1.6個請求，每個產生100個輸出詞元，所需總輸出為每秒160個詞元。分離前，每個請求以每秒10個詞元串流，解碼約10秒。這是解說算例，不是正式環境基準測試。[1]

穩態（Steady State）下，到達率乘以階段平均停留時間，即該階段平均活動請求數：1.6×10＝16個解碼請求。分散於四個混合節點後，每節點有四個活動序列。這是流量會計關係，不是通用批次大小規則。

理想化分離後，一個專用解碼器對四個活動序列，各以每秒40詞元串流。100詞元只需2.5秒；相同到達率下，1.6×2.5＝4個活動解碼請求。專用節點維持批次四，總輸出仍為每秒160詞元。

表面矛盾因此消失：單一請求串流快四倍，總輸出卻不變，因為更快的服務讓同時停留在解碼階段的請求變少。此結果不需要假設專用解碼器必須保留原本16個同時序列。

案例排除排隊、首詞元處理、傳輸、暖機與變異，目的是教讀者如何質疑基準測試。真實服務可能用改善來降低等待、符合更嚴格目標，或調整硬體配置；但實測證明前，不能先宣稱這些成果。

表7｜相同輸出，不同使用者體驗

衡量項目	混合工作節點	專用解碼
到達率（Arrival Rate）	每秒1.6個請求	每秒1.6個請求
每請求輸出	100個詞元	100個詞元
串流速度	每秒10個詞元	每秒40個詞元
解碼時間	10秒	2.5秒
活動解碼請求數	4個節點合計16個	1個節點共4個
總輸出	每秒160個詞元	每秒160個詞元

3.6 排程效益的條件式公式

令p為混合引擎用於預填充的時間比例。其他條件不變時，移除預填充中斷可帶來理想解碼排程倍率1 /（1−p）。75%時為4.0倍；50%時為2.0倍；10%時僅約1.11倍。[1]

公式用來理解一項機制，並非任意模型的效能預測。節點配置改變可能連帶改變批次、記憶體容納、通訊與排隊。原已將多數時間用於解碼的服務，單靠這項機制可回收的中斷時間有限。

預填充很快、解碼需求很高的系統，可能偏好混合設計，也可能因不同硬體、營運隔離或獨立擴充而採分離。相反地，長上下文與重複輸入處理可能造成高度預填充壓力。實際負載分布比通用架構口號更重要。

分塊預填充（Chunked Prefill）是另一方法。排程器把長輸入處理切成較小區塊，穿插解碼，避免單一工作占用引擎太久。這可能減少中斷，但區塊大小與排程政策仍有取捨。演講中的512詞元排程預算是案例設定，不是所有系統都該採用的數值。

經營者應問：在我的提示長度、輸出長度、突發流量及延遲目標下，哪種配置能讓每一元買到最佳有效服務？廠商應提供分布與達標率，不只挑最好看的平均數。

表8｜對原始預填充時間占比的敏感度

預填充時間占比 p	解碼占比 1−p	理想排程倍率
10%	90%	1.11×
25%	75%	1.33×
50%	50%	2.00×
75%	25%	4.00×

3.7 預填充與解碼分離必須付出的成本

分離預填充與解碼會產生交接（Handoff）。解碼節點需要輸入處理生成的狀態；移動KV快取消耗頻寬、時間與協調。僅看運算示意圖很吸引人的設計，加入傳輸、排程與排隊後可能失去優勢。

傳輸量不只是原始文字提示，而是模型衍生狀態；其大小取決於架構、序列長度、數值格式與平行化（Parallelism）。跨機器移動，與在緊密連接系統內保留狀態，可能是完全不同的工程問題。

Hang Yin 的Phala演講再加上一層複雜性：受保護環境改變可用通訊路徑，可能限制直接傳輸機制。矩陣乘法很快，不保證端到端模型服務同樣快。安全需求應從一開始納入架構決策。[15]

Baseten把PD分離、快取感知路由（Cache-aware Routing）及點對點狀態共享（Peer-to-peer State Sharing）結合。路由決定請求去向；重用避免重算；傳輸提供狀態；排程決定執行階段。正式環境成果屬於整套系統，不應自動歸因單一功能。[14]

有效實驗應以相同模型版本、精度、加速器總數、輸入 / 輸出分布與服務目標，比較混合與分離配置，並揭露快取政策、網路拓樸及暖機。若分離方案使用更多硬體，須揭露額外資源；若品質改變，亦須與效能結果並列。

表9｜PD部署決策紀錄

潛在效益	相關成本或條件
各階段獨立擴充	不平衡與節點閒置風險
更平順的解碼間隔	快取傳輸及交接延遲
硬體專業化配置	更複雜的配置與營運
工作負載隔離	獨立佇列與容量規劃
跨請求重用	正確的快取識別及生命週期管理

3.8 推論工作負載成長的四種方式

模型可以增加參數；請求可以延長上下文；服務可以增加同時使用者；代理可以增加嘗試或步驟。這些成長常被混在一起討論，實際上對基礎設施的需求不同。

更多參數通常需要更多權重儲存及計算，但關係取決於架構。混合專家模型（Mixture-of-Experts, MoE）可包含大量總參數，每個詞元卻只啟用部分。其餘參數仍須儲存、選取並於需要時提供。WiCi的SSD研究處理的正是容量與資料移動問題。[16]

更長上下文增加請求攜帶的資訊，可能提高預填充工作、狀態儲存及後續注意力成本，視模型與實作而定。更多同時序列會放大即時狀態量。單一使用者能放進記憶體的模型，未必能支援正式環境目標並行量。

更多代理步驟產生反覆推論及工具活動。小模型被多次呼叫，也可能消耗大量總資源。驗證增加成本，但可能提高有效成功率。因此，分母應是完成且通過驗證的任務或明確營運結果，而不只是模型呼叫次數。

成長問題因此轉為架構問題：哪項資源先用盡？可能是記憶體容量、頻寬、運算、互連（Interconnect）、排程軟體或外部工具。增加算術能力不能解決所有限制。記憶體分層、協調管理與代理演講的價值，就在指出不同瓶頸。

表10｜四個成長軸向

成長軸向	可能壓力	優先衡量項目
參數（Parameters）	權重與活動運算量	記憶體容納及每步成本
上下文長度（Context Length）	輸入處理與KV狀態	首詞元時間及狀態位元組數
並行量（Concurrency）	即時狀態與排程	負載下的尾端延遲（Tail Latency）
代理步驟（Agent Steps）	重複呼叫與工具作業	已驗證任務的時間與成本

3.9 Baseten——衡量整套優化

Tri Dao與Alexander Korte介紹優化的模型服務路徑與背後工程流程。此處Tri Dao為該場Baseten工程師，與同名FlashAttention研究者不同。效能剖析（Profiling）、歸因（Attribution）及驗證關卡（Validation Gates），連結效能工作、正確性、模型品質與負載行為。[14]



圖16｜Tri Dao與Alexander Korte介紹Baseten。作者會議攝影。[14]

演講使用不同時間尺度：以微秒計的合成層測試、解碼步驟時間、使用者詞元速度，以及即時應用的首詞元與完成時間。43微秒的單層結果不是完整回答延遲；推測式解碼一步也可能接受多個詞元。[14]

最重要的是嚴謹優化流程。更快的結果需要精確基準及保護有效行為的測試。快取感知路由、狀態共享與PD分離也說明，不能把正式環境成果自動歸因於單一功能。

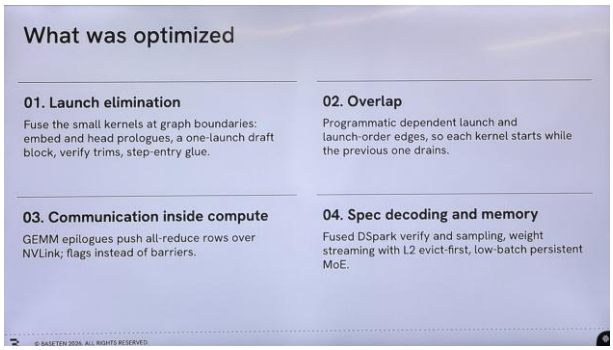


圖17｜Baseten優化堆疊（Optimization Stack）。作者會議攝影。[14]

投影片將工作分為消除啟動、重疊執行、把通訊融入運算，以及推測式解碼與記憶體。運算核心（Kernel）是由許多執行緒執行的GPU函式。[14]

微小操作可能將大量時間花在啟動、等待、資料移動或協調。單純加快算術，未必改善這些延遲。[14]

融合（Fusion）合併運算圖邊界的小操作；PDL讓符合條件的工作重疊。矩陣乘法（GEMM）的尾端處理（Epilogue）透過NVLink推送全歸約（All-reduce）資料，適當使用旗標。DSpark驗證與取樣被融合；L2優先驅逐提示（Evict-first Hints）管理串流權重；持續執行的MoE工作針對低批次情境。[14]

各技術降低不同開銷，必須協同運作。旗標仍須正確順序；快取提示改變保留政策，不改變模型知識或數值精度。[14]

流程是剖析、歸因、設立驗證關卡。Nsight Systems與PyTorch Profiler呈現步驟時間軸；Nsight Compute分析個別核心；CUDA Events衡量A / B比較時間。[14]

GPU追蹤圖的空白區間不會自己解釋原因。團隊須判斷是啟動開銷、資料相依、通訊，還是執行資源未充分利用。[14]

投影片要求把每個空檔或低利用率串流多處理器（Streaming Multiprocessor, SM）歸於特定核心或群組，先處理最大問題。SM是GPU運算單元之一。要求位元完全相同的變更須逐位元檢查；數值變更須通過對數機率KL檢查及推理模式GSM8K測試。[14]

先衡量再優化，並在相同條件下比較。否則暖機、上下文長度、輸出長度、並行量及模型設定，都可能在程式碼變更之外改變結果。[14]

三道保障為逐位元比較、模型品質檢查，以及當機 / 壓力測試。右側大量程式差異數字呈現審查量，不是已驗證的生產力。[14]

3.10 Traction Layer——可預期的服務本身就是產品

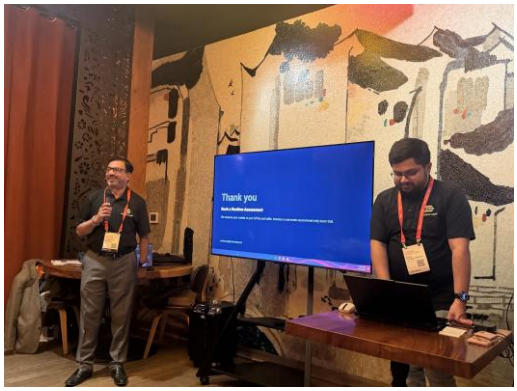


圖18｜Ghirardelli Square周邊活動的講者。作者會議攝影。[17]

在Ghirardelli Square的Dim Sum and Demo周邊聚會，Shankaran Sitarama與Rahul Kundapur Ramesh介紹Traction Layer支援的SGLang版本、強化建置（Hardened Builds）與營運控制。場地與峰會主舞台不同，產品訴求是更可預期的正式環境行為。[17]

企業經營者熟悉這個問題：快速變動的上游軟體（Upstream Software）難以持續採用。維持已知分支、回移修補（Backporting）與控制升級可減少干擾，但支援範圍與相容性必須清楚。

效能仍有取捨。平順的解碼串流可能需要延後其他工作或改變吞吐量。改善某項尾端延遲的控制器，應連同首詞元延遲、有效總輸出及失敗行為評估。穩定性可能值得顧客付費，但必須定義並證明，不能只靠「強化」一詞暗示。

投影片對照頻繁上游發布與較慢的企業變更核准。變更審查委員會（Change Board）在正式環境變更前評估營運風險。驗證（Qualification）是檢查實際應用仍能正確回答、延遲可接受、存取安全且能成功回復舊版，不只是確認軟體能啟動。[17]

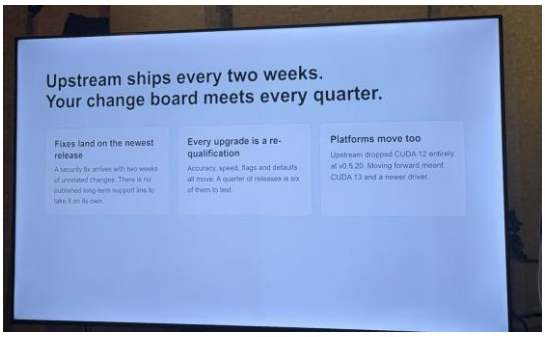


圖19｜上游發布頻率與企業核准流程。作者會議攝影。[17]

逐字紀錄提到CUDA變更。CUDA是輝達GPU運算平台；跨重大軟體環境可能需要相容程式庫與測試。這是講者舉例，不代表每次SGLang小更新都強迫升級CUDA。維護產品旨在減少重複驗證工作。[17]

回移修補（Backport）是把新程式碼的修正調整至舊的支援分支。周邊程式不同，未必能直接複製檔案，仍須修改與測試。投影片包含安全及正確性修補，排除新功能、預設值變更及其他行為改變。[17]

可用修門鎖比喻：替舊教室換掉壞鎖，不必重建學校；但全新教室配置就是較大變更。應問清楚包含哪些修補、多久提供、分支維護多久，以及無法安全回移時如何處理。[17]

接著進入投影片歸於Rahul Kundapur Ramesh的調校示範。問題是混合流量：短聊天、長提示與長答案共用引擎。系統整體產生很多詞元，個別使用者卻可能看到停住的回答。[17]

工程教訓是先分清不同時間指標，再重新連結。預填充、解碼、佇列與傳輸各解釋部分延遲。有效架構應以實測品質、可靠性與成本，符合完整負載需求。

4 為何必須看完整系統

上一章說明工作何時執行；本章追蹤工作存放何處、如何移動，從模型權重、快取注意力狀態，到記憶體分層（Memory Tiers）、加速器托盤、交換器與冷卻接頭。照片讓相依關係可見；軟體效率與實體基礎設施在可運作的配置中交會。

4.1 記憶體容量與資料移動塑造服務

記憶體容量（Capacity）回答能存多少資訊；頻寬（Bandwidth）回答移動多快；延遲（Latency）回答單次存取耗時；區域性（Locality）描述資料是否已靠近處理器。四者都重要，一項數字大不能代替其他項目。

Netpreme討論外部記憶體與選擇性存取的階層架構。其24 TB容量、原型頻寬與架構總頻寬屬不同範圍，不能合併宣稱外部記憶體等同封裝內高頻寬記憶體（HBM）。軟體必須在適當時機辨識並移動有用的資料子集。[18]

WiCi把類似原理用於消費級硬體上的大型混合專家模型（MoE）。固態硬碟（SSD）增加容量，但反覆讀取未快取資料可能限制輸出速度。依存取模式打包、避免不必要傳輸、管理重用，都是成果關鍵。單看龐大參數數量，無法判斷互動性。[16]

Xibo Sun介紹騰訊（Tencent）TACO，從介面、相容性與共享所有權處理狀態管理。KV容量增加本身不是吞吐量成果；服務仍需正確路由、同步、生命週期管理及回退路徑。[19]

有效推論容量，部分來自決定哪些資料不必移動或重新計算。記憶體軟體雖不如加速器醒目，卻能擴大可行負載、改善利用率（Utilization），或減少無謂硬體重複配置。經濟效益仍取決於工程成本及正式環境實測。

4.2 騰訊TACO——管理整個模型服務系統的狀態

Xibo Sun的騰訊演講強調分散式推論（Distributed Inference）的軟體約定。不同元件需要一致的表示及規則，才能查找、共享、搬移及淘汰狀態。實體佈局可不同，但相容範圍必須由介面界定。



圖20｜Xibo Sun介紹騰訊TACO。作者會議攝影。[19]

TACO的快取與平行化工作包含共享記憶體（Shared Memory）及狀態管理。營運測試應問：真實負載下能否正確重用與移動狀態，而不只是能否儲存更多。[19]

標準化邊界可降低整合成本，讓替代後端更易使用；也能辨識不支援情境，予以拒絕或導向回退路徑。投影片CP8案例等容量增加，沒有實測負載結果前，不能描述成吞吐量增加。

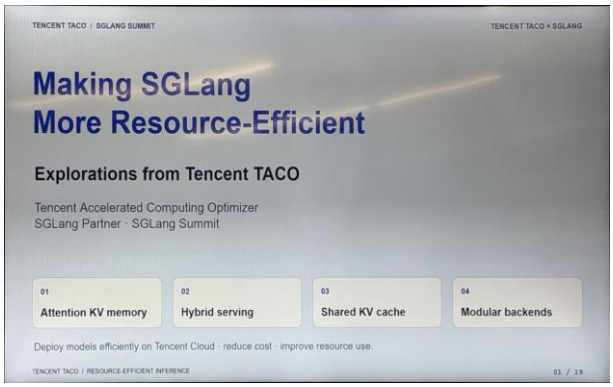


圖21｜騰訊TACO資源管理。作者會議攝影。[19]

騰訊TACO提出四種改善模型執行的方法：減少重複記憶體、彈性使用服務容量、共享已快取工作、簡化後端整合。這是關於推論，也就是以訓練完成模型產生答案的演講。[19]

可想像學校圖書館：若書籍不必要地重複、部分書桌閒置、學生反覆搬同一本書，增加書桌不能解決所有問題。演講檢視人工智慧服務內類似浪費。[19]

四項工作分屬不同層級：分散引擎內的注意力記憶體、預填充與解碼節點間路由、跨機器快取管理，以及軟體模組介面。後續投影片逐一連結具體機制。[19]

資源效率（Resource Efficiency）是以既有資源完成更多有用工作。投影片同時包含已實作實驗與未來方向，本報告保留兩者區別。[19]

路線圖包含FSCP、P+PD、FlexKV與RadixShmem，以及框架 / 後端解耦（Framework–Backend Decoupling）。FSCP處理重複KV儲存；P+PD處理不均勻需求；FlexKV處理快取重用；後端介面處理整合相依性。[19]

單一利用率百分比掩蓋不同問題。GPU可能運算能力有餘，記憶體卻不足；解碼伺服器忙於產生答案，CPU記憶體仍可能有空間。各資源需要各自政策。[19]

四節是互補提案，不是四個可相乘的效能倍率。利益可能重疊，節省記憶體也可能增加通訊或排程工作。[19]

先辨認受限資源，再評估對應技術。記憶體容量、傳輸成本與排程彈性需要分別衡量。[19]

4.3 Netpreme——延伸記憶體階層



圖22｜Nikita Lazarev解說記憶體分層（Memory Tiering）。作者會議攝影。[18]

Nikita Lazarev的案例從限制出發：高速封裝內記憶體珍貴，難以無限擴充。外部記憶體提供更多容量，但系統須夠快地移動正確資訊，才能維持實用效能。

演講結合記憶體階層、選擇性存取與資料移動軟體。24 TB容量、單GPU原型頻寬及更廣架構頻寬屬不同測量，應與各自配置及範圍連結。較大的記憶體池不等於同容量本地HBM。

機會在於增加有效狀態容量，又不必為不必要的重複運算設備付費；風險是存取模式與傳輸抵銷效益。商業評估須涵蓋端到端負載、快取行為、網路競爭與整合工作。最有力比較是在固定品質及延遲目標下的有效服務。



圖23｜超越HBM的記憶體分層。作者會議攝影。[18]

金字塔頂端是小容量靜態隨機存取記憶體（SRAM），下方HBM標示288 GB與20 TB/s，再以3.6 TB/s連結接往24 TB外部池。這些是架構標示。[18]

並非每個位元組每刻都需要最快記憶體。外部容量可保存較長歷史，高速記憶體保存活動工作集（Working Set）。[18]

軟體程式庫管理KV快取、稀疏注意力（Sparse Attention）、Engrams及專家卸載（Expert Offloading）。統一記憶體（Unified Memory）描述程式設計與管理目標，不代表各位置存取時間相同。[18]

十進位計算下，24 TB約為288 GB的83倍，但只是容量比。不能把此投影片當成已出貨B300的頻寬規格。[18]

密集注意力（Dense Attention）在HBM讀取並保存完整快取；一般稀疏注意力只讀部分紀錄，卻可能仍把全部資料放在HBM。HiSparse加入分層，使HBM只保留熱門資料集（Hot Set）。[18]

讀取較少紀錄不會自動釋放容量；後續查詢可能選不同紀錄，因此完整歷史仍須可取得。[18]

雙向箭頭透過PCIe或晶片間連結，連接HBM與CPU動態隨機存取記憶體（DRAM）。快取未命中（Cache Miss）時取回所需資料。KV快取保存的是模型中間表示，不是字面上的文字頁面。[18]

最高4.7倍解碼效能主張，屬所引用HiSparse研究的測試情境，與Netpreme後續2.4倍比較不是同一結果。[18]

預填充GPU處理初始提示並發布完整KV快取；解碼GPU用熱門KV緩衝區產生新詞元；X Mem保存共用24 TB快取池。[18]

將大型歷史與活動緩衝區分開，可能支援更多請求，而不要求每張解碼GPU在本地保存每筆快取。[18]

4.4 WiCi InferSSD——以頻寬取捨換取容量



圖24｜Zili Meng介紹WiCi InferSSD。作者會議攝影。[16]

Zili Meng探討如何以SSD容量，在消費級硬體執行極大型MoE模型。系統必須有效提供被選中專家的權重，而不是一次把所有參數載入稀缺高速記憶體。

InferSSD說明資料佈局、明確讀取、記憶體預算，以及不同預填充／解碼政策。照片中每秒7.27詞元屬於該展示配置，不能與其他產品說明規格拼接成更漂亮的綜合結果。[16]

方法可能擴大本機可執行範圍，但買方仍應評估首詞元時間、持續輸出、儲存行為及完整持有成本。「不額外量化（No Additional Quantization）」不自動代表完整FP16權重；必須交代模型格式與基準。

展示峰值為：顯示記憶體（VRAM）1,792 GB/s、主記憶體（RAM）51.2 GB/s、SSD 14.8 GB/s。前兩者相對SSD約為121倍與3.46倍。[16]

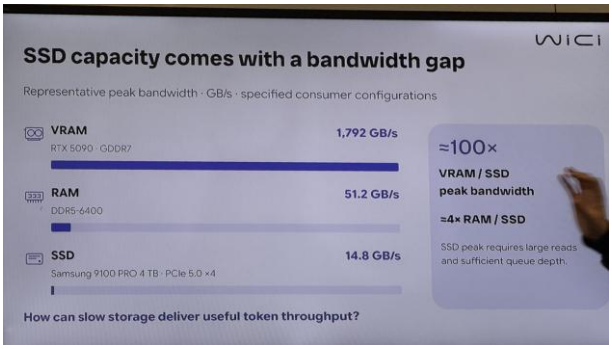


圖25｜SSD容量及其頻寬落差。作者會議攝影。[16]

高速GPU仍可能因等待權重而閒置。裝置峰值頻寬也不同於有效端到端頻寬。[16]

示例：以14.8 GB/s讀10 GB，未計其他成本就至少約0.68秒。較大連續讀取、足夠待處理讀取及重用有所幫助；RAM頻寬則取決於通道配置。[16]

技術堆疊由推論引擎及運算子，經專家執行環境、檔案系統、NVMe驅動程式，到SSD控制器。[16]

各層都可能增加等待或複製。改善單一階段後，吞吐量仍受剩餘最慢階段限制。[16]

執行環境把所選專家映射至儲存位置，讀取缺少資料，並安排CPU或GPU工作。檔案系統與驅動協同設計，是比開源執行環境本身更廣的系統工程。[16]

投影片大學標誌顯示研究背景，不代表公司所有權或財務投資。[16]

一般推論期間，權重固定，啟動值（Activations）與KV狀態則會改變。「唯讀」指權重，不是整個程式。[16]

穩定權重可預先佈局、快取並分散到多顆硬碟。快閃記憶體讀取消耗的寫入耐久度，通常遠低於寫入／抹除循環。[16]

若通道與控制器支援，四顆硬碟可保存不同模型部分並同時讀取；若共享狹窄連結，瓶頸仍在該處。[16]

讀寫一致性不是HBM或DDR容量難擴充的唯一原因；封裝、功率、成本與互連也重要。[16]

直接GGUF索引定位既有權重區段；可選的打包功能將同一專家的gate、up與down矩陣放在一起。[16]

4.5 輝達——從基礎設施到晶片解讀機架

後視圖改變敘事。正面是成排運算設備；背面呈現讓設備可用的連接與維修空間。這解釋基礎設施為何屬於餐廳營運平台報告的續篇：部署取決於周邊系統，而不只是取得能力強的晶片。[7, 2]

後方標記1為上方大型管接頭；2為密集互連與維修區；3為纜線引導結構。水平黑色線路並非全是水管；外觀不能證明隱藏電氣額定值、閥門配置或確切機架世代。[7]

實務問題是維修可及性。技師需要空間隔離電源與冷卻液、接觸連接點並取出托盤，而不傷及相鄰纜線。平面圖放得下的機架，仍可能難維修。這連結AI設備、可用建築容量、維修時間及可靠服務的經濟效益。[7]



圖26｜輝達機架另一側。作者攝影，附編號標註。[7]

把正面視為不同工作的分布圖



圖27｜正面各區執行不同工作。作者攝影，附編號標註。[7]

標記1——上方網路設備。可見機箱似網路裝置；確切管理角色須對照物料清單（Bill of Materials, BOM）。網路設備不是另一個GPU托盤。

標記2——類似電源層架（Power Shelf）的區域。重複小模組符合供電設備特徵；把每個模組算成GPU，會混淆供電與運算。

標記3與5——類似運算托盤的區域。可抽換單元提供CPU與GPU工作的伺服器結構；展示照片不能單獨核實實裝數量或通電狀態。

標記4——類似交換器托盤的區域，工作是移動加速器間資料。即使外觀像運算伺服器，通訊能力仍是不同系統元件。此機架照片不足以確認確切產品世代。[7]

我的解讀是：買方應先指定工作負載及服務水準，再比較機架。硬體、網路、冷卻與支援須以可運作配置整體報價；展示或元件數量都不是完整採購報價。[7]

GB300內部——運算、電力與熱



圖28｜附識別牌的GB300運算托盤。作者攝影，附編號標註。[7]

標牌可讀為GB300 NVL72 Compute Tray，比靠近贊助商標誌更能確認身分。輝達參考架構的標準托盤為兩顆Grace CPU與四顆Blackwell Ultra GPU。數量來自技術文件；照片顯示的是覆蓋晶片的冷卻組件。[7]

標記1、2為兩個運算板區域。每個標準板結合一顆Grace CPU與兩顆GPU；CPU協調主機工作，GPU執行平行模型計算。可見冷卻組件不是底下每顆晶片的直接照片。[7]

標記3為風扇列，4為配電與纜線。同機箱不同組件冷卻需求不同，因此液冷與風扇可並存。托盤是伺服器組成單元，不是整座NVL72機架，仍依賴相容交換、配電及液體接頭。[7]



圖29｜GB300電線、液冷接頭與風扇。作者攝影，附編號標註。[7]

追蹤連接而非只看顏色：第二張照片中，1是粗紅黑電導線，2是金屬液體接頭，3是風扇，4是覆蓋軟管或冷卻路徑；亦可見編織訊號線束。液體接頭上的紅藍環，不會讓它變成電氣連接。[7]

各系統失敗方式不同：資料連接可能降頻寬或產生錯誤；冷卻受阻可能過熱；電氣故障可能中斷服務。整體運轉前，應分別檢查網路、液壓與電氣系統，正如模型服務基準須分別驗證品質、延遲與可靠性。[7]

經營者重視的是持續完成有用工作。機架成本須涵蓋安裝、冷卻能力、更換流程及支援；峰值算術指標無法回答這些問題。[7]

Rubin——運算托盤與交換器托盤是不同採購項目



圖30｜Rubin運算托盤及覆蓋的處理區。作者攝影，附編號標註。[7]

清楚標牌與廣角照片區分Rubin運算托盤與交換器托盤。標準運算托盤含兩顆Vera CPU及四顆Rubin GPU。黑蓋與冷板（Cold Plates）遮住封裝邊界，因此編號區域不是四顆逐一裸露的GPU封裝。[7]

標記1至4為主要覆蓋組件位置，5為中央結構及維修區。大致對稱的佈局有助理解模組化，但有把手不代表能在通電時更換。[7]

商業問題是：更換會中斷單一元件、整個托盤，還是更廣工作？維修程序與可替換單元很重要，因為硬體維護會影響推論服務。[7]

交換器照片呈現另一種功能。標記1是管理板與小型儲存模組區，2是紅藍環液體接頭，3是參考設計中四顆NVSwitch特殊應用積體電路（ASICs）的冷卻組件。這些專用晶片搬移資料，不執行機架AI GPU的角色。[7]

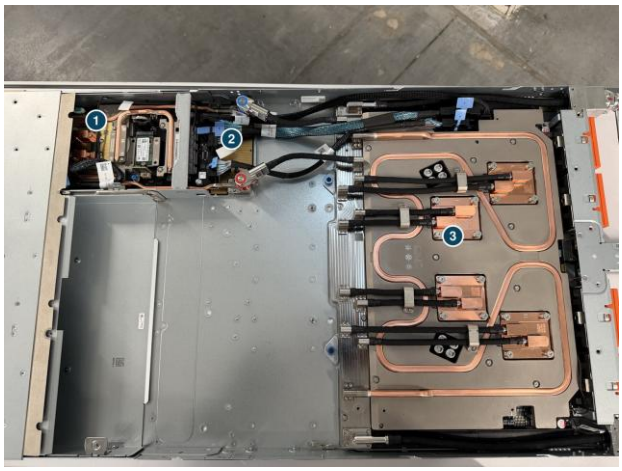


圖31｜Rubin交換器托盤的管理與冷卻區。作者攝影，附編號標註。[7]

交換器仍需要韌體（Firmware）、開機支援、監測與本地控制。內有儲存裝置與電池，不代表它是運算托盤。較廣照片將M.2模組與液體接頭定位在Rubin交換器托盤，釐清孤立特寫的歧義。[7]

互連是實際模型效能的一部分。只買加速器而無適當通訊網路（Communication Fabric），無法重現整合機架行為。應在系統層級比較，明列模型、精度、並行量及服務目標。[7]

Vera——八顆CPU封裝不是八張GPU



圖32｜八個處理器位置編號的Vera CPU托盤。作者攝影，附編號標註。[7]

展示標籤為Vera CPU Compute Tray，可見八個大型裸露處理器封裝。標記定位的是封裝，不是八顆GPU加速器或八個CPU核心；鄰近Rubin運算托盤的GPU數量是另一問題。[7]

封裝（Package）是實體處理器元件；核心（Core）是內部執行資源。若實裝為文件中的88核心配置，八顆CPU共704核心。此算式不能證明展品時脈、記憶體配置或運作狀態。[7]

解讀硬體行銷時須保留區別。CPU核心、GPU裝置、記憶體容量與浮點運算量衡量不同事物，都不能直接替代模型服務可支援的同時顧客數。[7]

特寫顯示兩顆處理器封裝及相鄰記憶體模組陣列，屬同一八CPU系統，不能額外再加兩顆。實裝容量取決於零件與插槽配置，不能靠數照片中的黑色方塊決定。[7]

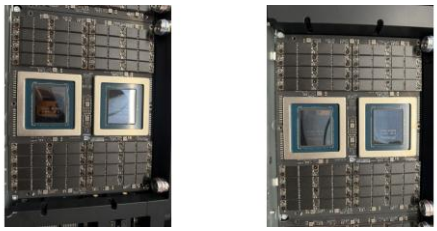


圖33｜Vera CPU封裝與周邊記憶體陣列。作者攝影，附編號標註。[7]

推論研究的有效鏈條，是從負載走向配置：模型要做什麼、哪些狀態須留在記憶體、資料移動多快、完整系統交付服務時消耗什麼？機架照片提供實體情境，不能取代驗收測試。[7]

下一次現場訪談問題：提供哪個確切配置？報價包含哪些元件？在何種同時負載下衡量哪些品質與回應時間目標？元件故障或計畫維護時會如何？[7]

4.6 AMD——從加速器板卡到完整伺服器

AMD照片區分八加速器平台與另一台開蓋機架式伺服器，不能拼成虛構配置。展牌將加速器板卡識別為MI350X；另台銀色機箱內確切CPU與GPU配置仍未確認。[6]



圖34｜共用底板上的八個AMD加速器模組。作者攝影，附編號標註。[6]

兩列各四個，合計八個大型黑色組件，是加速器上方散熱器；GPU封裝及高頻寬記憶體在下方。藍色底板提供共用電力、訊號與管理連接。完整伺服器還需主機處理器板、系統記憶體、儲存與網路。[6]

展品識別與AMD文件提供規格背景：每GPU具288 GB HBM，八GPU合計2,304 GB，以展示單位約2.3 TB。這是本地記憶體加總，不保證每個位元組都有一致存取速度。額定頻寬與功率也須和應用實測分開。[6]

採購問題是所選模型如何使用資源。多個模型副本（Replicas）與跨加速器切分模型，有不同通訊及延遲需求。總記憶體不足以決定最佳服務策略，照片也未證明八模組都在運行。[6]



圖35｜MI350X規格牌確認產品系列。作者攝影，附編號標註。[6]

AMD展品內部——記憶體、儲存與未識別處理區

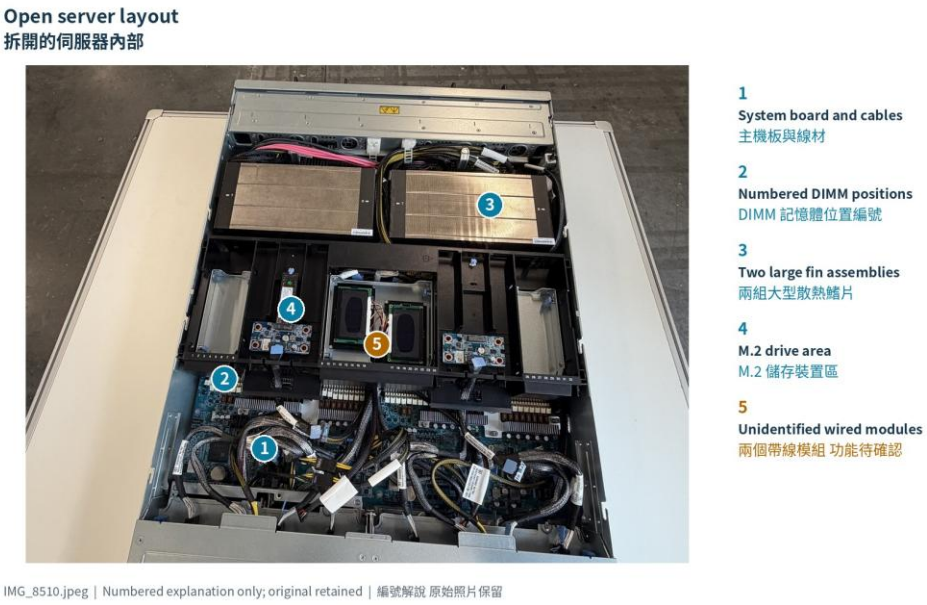


圖36｜銀色伺服器開蓋內部。作者攝影，附編號標註。[6]

藍色印刷電路板為系統主板。重複插槽及白色卡扣標示雙列直插記憶體模組（DIMM）位置；編號至32代表位置，不是32顆CPU或GPU。黑色隔板引導氣流，編織線束承載電力或訊號。兩個大型銅色鰭片組件散去底下元件熱量；尺寸不能證明CPU型號或數量。[6]

帶白標籤的狹長模組是M.2固態硬碟；M.2描述實體規格。中央兩個接線模組仍未分類，須取得主板手冊或展商確認，才能賦予處理器角色。[6]

伺服器管理標籤也不是CPU型號。基板管理控制器（Baseboard Management Controller, BMC）是執行遠端電源控制與健康監測的小型管理電腦。資產識別碼有助設備管理，不揭露GPU數量或峰值算力。[6]

具體追問應是：索取伺服器型號牌、已配置元件清單，以及各處理器與加速器位置圖。保留照片可見內容與配置表才能證明內容的區別。[6]



圖37｜M.2儲存裝置及兩個尚未識別的接線模組。作者攝影，附編號標註。[6]

AMD細節——小型支援元件讓大型加速器可用



圖38｜主板接線、記憶體卡扣與局部冷卻。作者攝影，附編號標註。[6]

主板特寫呈現處理器周邊工作。白色卡扣固定記憶體；小銀色鰭片冷卻局部電路，可能是控制器或電源調節。鈕扣電池支援即時時鐘等功能，不供應整台伺服器。纜線套保護線束；接頭功能仍需主板文件確認。[6]

大型黑塊是冷卻組件，鰭片增加表面積，讓伺服器氣流帶走熱。路徑從半導體經導熱介面材料（Thermal-interface Materials）至散熱器，再至空氣。每個黑塊上看不到小風扇，不代表不能用於氣冷伺服器。[6]

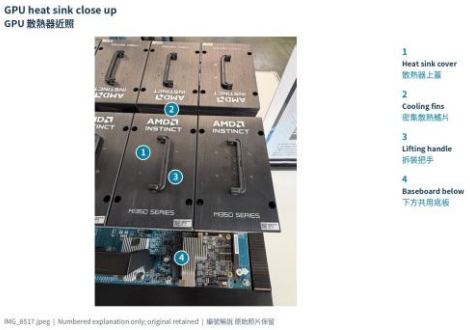


圖39｜AMD加速器上方散熱組件。作者攝影，附編號標註。[6]

MI350 SERIES是系列名稱，旁邊MI350X標牌提供更精確身分。特寫只涵蓋部分陣列，不能取代廣角照片的八模組計數，也未直接露出GPU晶粒（Die）或HBM。[6]

與SGLang的連結在營運層面：儲存保存模型檔案；主機處理器及系統記憶體協調載入與請求；加速器記憶體保存模型執行資料。ROCm、框架及服務層須匹配實際裝置與版本。高效軟體可改善利用率，不能增加實體GPU數量。[6]

第二集結論：第一集餐廳據點與此處伺服器共享一項商業限制——整套營運系統都須有效。買方需要受支援配置、實測服務品質與全系統成本。照片能證明元件及情境，無法補足缺少的基準測試或配置清單。[2, 6]

4.7 Uni³Sim——採購前先測試系統設計

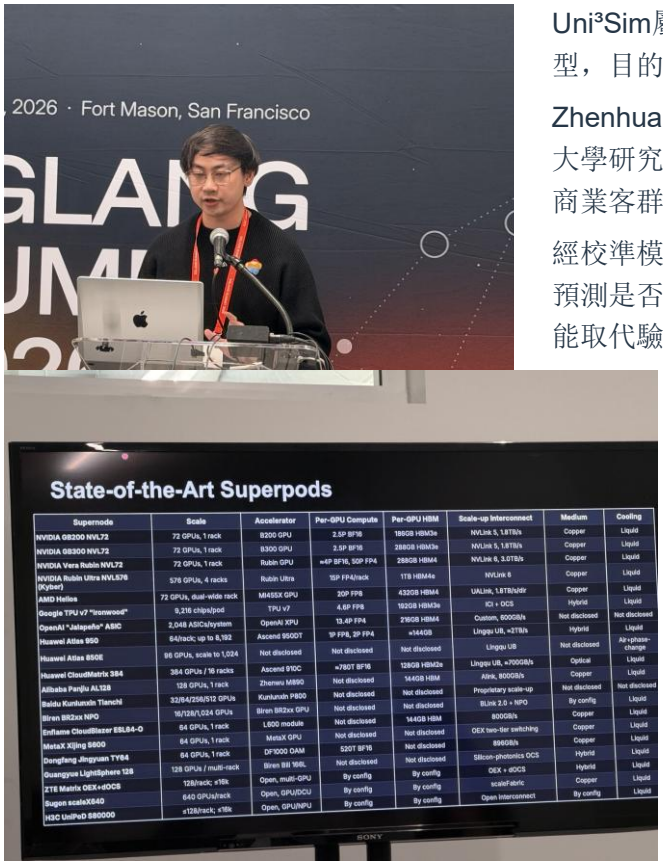


圖41｜Uni³Sim系統配置。作者會議攝影。[20]

Uni³Sim屬AI基礎設施設計，連結排程行為與運算、記憶體及互連模型，目的是探討系統選擇如何互相影響，而非孤立優化單一元件。

Zhenhua Zhu將Uni³Sim介紹為與清華大學深圳國際研究生院相關的大學研究專案。證據支持研究身分，未證明其為獨立募資公司或已有商業客群。我未稽核其程式碼儲存庫（Repository）。[20]

經校準模擬器（Simulator）可在採購前淘汰不佳配置；價值取決於預測是否符合實測，以及準確性是否超出校準案例。視覺有說服力不能取代驗證；預測與實測延遲須採一致單位。

表格比較加速器、規模、記憶體、連結、傳輸媒介與冷卻，呈現龐大設計空間。[20]

各欄須連同單位閱讀。BF16、FP8與FP4效能不能直接互換；每晶片與每機架數字不同。投影片在每GPU欄名下含一項每機架數值；產品規格與供貨狀況未獨立核實。[20]

電子設計自動化（Electronic Design Automation, EDA）協助設計與驗證晶片；系統級EDA擴展至封裝、板卡、機架與網路。[20]

局部最優元件可能組成不佳系統。示例中五種晶片×四種記憶體設計×三種網路×六種部署政策，共360種組合。模擬可在實體測試前篩選。[20]

請求進入批次，模型與平行化設定產生核心任務；執行改變記憶體占用與經過時間，再回饋排程器。執行越久，累積請求可能越多。H2D指主機到裝置，D2H相反，D2D為裝置到裝置。[20]

運算核心是可在裝置上執行的程式，例如矩陣乘法；硬體峰值規格不能描述每種負載。[20]

先估算運算量、執行波次、快取未命中、記憶體流量及並行量，再用硬體方程式換算服務時間。循序工作相加；獨立工作在資源允許時可重疊。白箱（White-box）表示假設可供檢視。[20]

團隊報告1,027案例：221個GEMM、334個注意力、176個MoE及296個向量案例，衡量時間誤差，不衡量答案正確性。[20]

Uni³Sim平均絕對百分比誤差（Mean Absolute Percentage Error, MAPE）為A800的7.6%、MetaX C500的7.1%、Ascend 910B的12.9%。散點圖橫軸為實測延遲，縱軸為預測延遲；對角線代表一致，對數座標涵蓋不同尺度。[20]

機架與記憶體案例改變容量的意義。裝置數只是庫存指標；記憶體、資料移動、電力、冷卻、排程與軟體共同運作後，才形成有效推論容量。這才是應用團隊最終要買或營運的系統。

5 從快速詞元到有用工作

使用者要的是完成的結果，模型服務引擎只提供其中一部分。其他工作可能包括搜尋、工具呼叫、安全決策、驗證或人工審查。以下案例說明快速模型與有效應用需要不同衡量，也說明任務邊界的重要性。

5.1 衡量完整工作流程（Workflow）

Arcee的《快速詞元，緩慢代理（Fast Tokens, Slow Agents）》對吞吐量行銷形成重要補充。代理（Agent）可能檢索資訊、呼叫模型、執行工具、觀察結果，再重複。模型只是序列中的一個元件。[12]

假設研究流程60秒：15秒產生文字、30秒等待搜尋與工具、15秒驗證及組裝結果。生成快三倍後，只需5秒，總時間變50秒，即1.2倍加速，而非3倍。這是作者教學算例，不是Arcee公布的基準。

這不代表推論優化不重要，而是釐清利益出現何處。生成占主導時效益很大；工具路徑占主導時，下一步可能是快取、平行檢索、更佳查詢、減少重試或簡化任務。答案須丟棄重做時，品質失敗也可能主導成本。

顧客分析亦然。語言模型很快摘要抱怨，專案卻可能卡在分散的顧客識別碼、缺漏同意紀錄或不一致交易歷史。改善模型不會自動核對這些資料。應衡量每一階段，讓投資對準真實限制。

研究應報告兩層效能：元件結果與商業任務結果。前者讓工程師重現技術改善，後者告訴買方改善是否重要。可信的推論經濟分析需要兩者。

表11｜端到端任務示範

階段	改善前	生成快三倍後
模型生成（Model Generation）	15秒	5秒
搜尋與工具	30秒	30秒
驗證與組裝	15秒	15秒
合計	60秒	50秒

5.2 Arcee——代理仍有工作要完成

Lucas Atkins將模型放進更大的軟體系統。提示建構、檢索、工具、上下文重用、取樣（Sampling）、評估及環境設計，都可能決定任務時間與品質。

因此買方須重新解讀模型效能。更快詞元串流可能有價值，卻不能消除工具延遲或修正不佳流程。持續增加上下文還可能增加輸入工作，讓失敗更難診斷。

應把代理系統當完整流程剖析，衡量時間花在哪、為何重試、哪些輸出通過驗證。這直接關係餐飲案例：模型應協助可重現分析流程，而非以流暢敘事掩飾缺漏資料或未定義的決策規則。



圖42｜講台上的Lucas Atkins。作者會議攝影。[12]

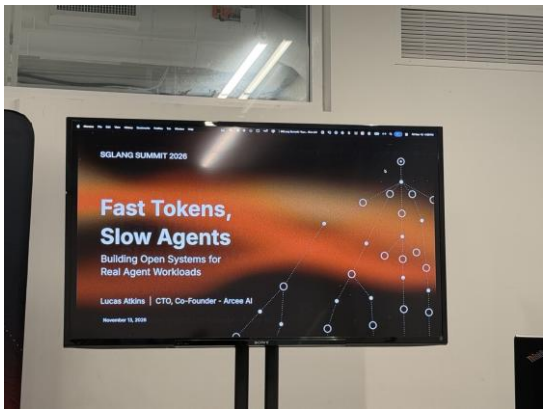


圖43｜快速詞元，緩慢代理（Fast Tokens, Slow Agents）。作者會議攝影。[12]

投影片比較「生成→閱讀→決策」與「生成→驗證→執行」，輸出使用者從人轉為程式。[12]

人需要理解時間；輸出超過舒適閱讀速度後，再加速的價值可能遞減。機器則可在指令有效時立即執行。[12]

試算表工作中，人閱讀指示再點選選單；代理驗證結構化指令並呼叫函式。驗證須檢查意義與權限，不只是語法。[12]

機器使用輸出可降低人類閱讀瓶頸，但網路、啟動及執行延遲仍存在。[12]

兩條長條均包含首詞元時間（TTFT）、可用輸出生成及工具執行；解碼產生輸出詞元。[12]

工具往往不能執行半個指令，需要完整檔名、參數或修補。第一個可見字不一定已是可用工作。[12]

縮短橘色生成區段，會讓虛線交接點提前。藍色執行區段提早開始，但長度不變。投影片未提供數值時間刻度。[12]

T(s)＝D/s＋R。D是原解碼時間，s是加速倍率，R是此項優化未縮短的時間。[12]

生成越快，排隊、沙箱（Sandbox）準備、檔案作業與工具就占總工作更大比例。[12]

5.3 Palo Alto Networks——太晚的安全答案可能失去作用

Matthew Schulz與Zhemin Su的演講檢視嚴格延遲目標下的安全推論。模型服務的決策若太晚，營運價值可能很低，因此符合目標的輸出比單獨追求高平均吞吐量更重要。[21]

展示比較把第99百分位延遲（p99 Latency）從53.1降至44.2毫秒，約改善16.8%；符合目標的同時客戶端數從2增至3。兩者分母不同，均不能把整個服務的請求量歸到單張GPU。[21]

評估延遲敏感AI，須指定負載、硬體、模型與通過標準，並同時計算失敗。投影片與口述對模型大小分別提到7B及8B，差異仍未釐清。[21]

示例：10,000請求的p99接近50毫秒時，約100筆可能落在較慢的1%尾端，視百分位算法與同值情況而定。安全關鍵企業仍需處理延遲請求的政策；此數字不是演講實測。[21]

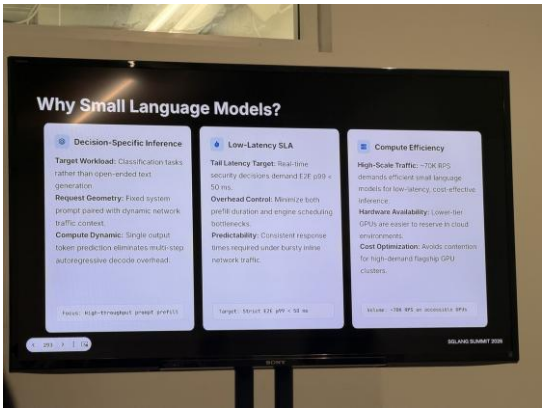


圖44｜為何使用語言模型做安全推論。作者會議攝影。[21]

預填充處理輸入，產生選擇首詞元所需資訊。單詞元分類避免後續詞元的重複自回歸解碼，卻不消除輸入注意力、輸出選擇及所有排程工作。後續排程器錯誤說明，單詞元任務也可能產生多餘執行。[21]

服務水準目標（Service-level Objective, SLO）是營運目標；服務水準協議（Service-level Agreement, SLA）通常可能具有合約意義。口述用SLO，投影片寫低延遲SLA。未提供合約條款，因此本報告將50毫秒稱為目標。[21]

講者描述支援不同模型家族與推論框架，涵蓋線上安全決策及離線批次處理。口述中不清楚的名稱保留未確認，不任意對應產品。[21]

架構討論涵蓋解碼器、編碼器模型、多種服務框架及異質加速器（Heterogeneous Accelerators）。原則是為模型選相容軟硬體，不代表每個框架都支援每個裝置上的每個模型；不清楚口述仍保留未確認。[21]

SGLang是測試語言模型的運算服務層；Ray Serve與Kubernetes處理較高層的應用及基礎設施。慢請求可能來自等待副本、CPU前處理、GPU執行或回傳客戶端。優化一個核心不能消除其他全部延遲。[21]

5.4 Phala——隱私必須貫穿執行路徑



圖45｜Hang Yin介紹機密推論（Confidential Inference）。作者會議攝影。[15]

Phala討論在執行中保護資料，並向客戶端提供受保護環境的證據。Hang Yin說明驗證證明（Attestation）、客戶端查驗，以及不暴露私密內容下觀察與調校系統的困難。[15]

效能細節顯示安全與服務架構互相影響。矩陣運算可保持快速，傳輸或CPU排程卻成瓶頸。直接GPU通訊機制是否可用，取決於演講中的保護配置。

機密運算（Confidential Computing）可能擴大客戶願意部署的負載，但不是模型正確性或資料治理的通用保證。信任模型、軟體身分、金鑰處理與仍暴露的中繼資料都須明列。標題中的每日2,000億詞元仍屬廠商規模主張。

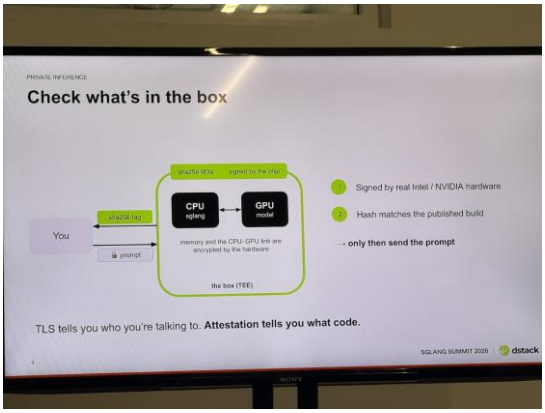


圖46｜機密執行與驗證證明（Attestation）。作者會議攝影。[15]

驗證證明提供受保護環境及實測軟體狀態的證據。投影片顯示在傳送提示前，先回傳硬體簽署的測量值。[15]

供應商可能展示隱私標誌，實際卻沒執行預期配置；查驗讓這項主張可測試。[15]

客戶端檢查證據真實性、可接受測量值，以及證據是否連結到接收機密的端點（Endpoint）。SHA-256雜湊（Hash）是數位指紋，不是加密，也不是完整程式碼審查。[15]

有效測量可在驗證政策下確認身分與配置，但不能證明應用無錯誤、不會洩漏輸出，或AI答案正確。[15]

下一步連結公開原始碼與伺服器上實測建置，補充實際部署軟體的證據。[15]

公開程式碼不能證明服務執行的就是該程式碼，也可能部署修改版。[15]

可重現建置（Reproducible Build）希望另一人使用相同輸入與工具，得到相同產物。驗證者比較預期建置身分與驗證證明中的身分。[15]

證據鏈包括原始碼審查、受控建置、實測部署及已驗證連線。只查網站聲明或只看儲存庫，都有缺口。[15]

正式服務有不斷變動的工作節點群。節點是執行推論的程序或機器，不一定各對應一張GPU。[15]

節點會重啟、擴充或遷移。昨天驗證一台機器，不能證明今天接收請求的機器身分。[15]

圖示追問哪些節點被查驗。驗證須隨成員變動更新，不能只是一次人工工作。[15]

5.5 Core Automation——讓研究實驗更容易執行



圖47｜Francis Zhang解說研究基礎設施。作者會議攝影。[13]

Francis Zhang將推論視為行為須可預期的研究工具，檢視工作階段識別（Session Identity）、快取行為與獨立記憶體池，以組織工作並避免實驗或請求類別間不必要互相干擾。[13]

只有精確認定可重用的先前狀態，快取才有用。上下文、模型配置或工作階段處理的隱藏變更，都可能破壞表面可比的測試。研究者可能同樣重視可重現性與峰值速度。

推論基礎設施應讓實驗控制條件可見。演講沒有證明架構的通用實測加速倍率；其意義在嚴謹管理狀態與請求邊界，提高研究結果的可解讀性。[13]

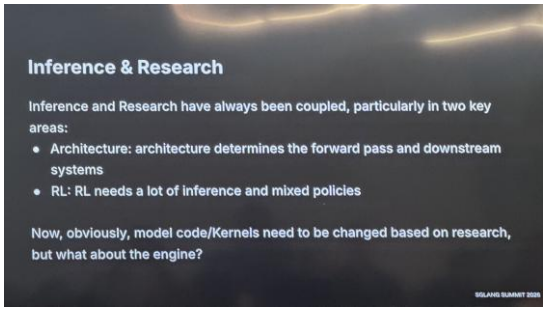


圖48｜推論與研究。作者會議攝影。[13]

投影片指出兩項連結：架構決定前向傳播（Forward Pass），強化學習需要大量推論。架構是模型內部計算安排；強化學習（RL）利用結果與獎勵（Rewards）改善行為。[13]

改變注意力機制，可能改變狀態與記憶體需求；若RL需要數千次回答嘗試，生成太慢會延遲下一次學習。引擎因此影響哪些實驗實際可行。[13]

簡化RL循環四步：模型嘗試任務、評估器或環境提供結果、訓練更新模型、更新模型再嘗試。嘗試常稱軌跡生成（Rollout）；推論產生經驗，之後訓練改變權重。[13]

技術背景：詞元（Token）是文字片段，可能是一個字詞或部分字詞。對數機率（Log Probability）將多個詞元機率的乘積轉成較易處理的加總。它是模型分數，不保證答案正確。[13]

示例：數學家教生成十種解法，檢查後將回饋用於後續學習更新。即使沒有顧客在聊天視窗等待，生成及評分仍消耗推論。[13]

研究基礎設施須讓反覆模型執行可靠且合理高效。「混合策略（Mixed Policies）」亦要求追蹤哪個模型版本產生哪些資料；演講未指定確切策略混合演算法。[13]

抽象化（Abstraction）是隱藏細節的實用介面；隔離（Isolation）避免不相關變更迫使研究者理解全系統。描述新注意力方法，不應同時要求重設所有記憶體配置規則。[13]

不同應用需要不同驗收測試：代理完成有效任務；安全服務在決策窗口內回答；機密推論維持信任邊界；研究引擎支援可重現實驗。更快詞元可協助，但不能取代這些測試。

6 能搜尋、學習與驗證的推論

推論不只產生最終答案，還可產生待檢查的證明、動作、候選圖像或研究假說，再形成下次更新的回饋。額外運算帶來更可靠結果時，商業機會擴大；若只是重複未驗證假設，機會便削弱。

6.1 SGLang與Miles——模型服務支援學習循環

開場討論把模型服務與後訓練（Post-training）放進同一工程循環。強化學習呼叫推論生成經驗；訓練系統評估、更新權重，再把新模型版本交回生成端。[3]

此處有不同組織關係。SGLang是LMSYS Org託管的開源專案；Miles是在RadixArk GitHub組織維護的開源框架，公開承認源自slime；RadixArk圍繞兩者提供商業基礎設施。這是專案、維護者及商業服務的關係，不是兩家名為SGLang與Miles的公司合併。[3]

整合可減少相容與營運工作，但買方仍須檢視支援配置、退出路徑及學習循環品質。開源不代表部署免費，託管服務（Managed Service）也不能免除獨立評估。



圖49｜Ying Sheng，峰會官方肖像。峰會官方資料。[3]



圖50｜模型服務與強化學習回饋。GII解說圖。[3]

維護者技術決策與贊助商商業利益須區分。開源讓使用者可檢視實作並採替代營運路徑；封裝、維護與託管仍可是有價值的付費工作。[3]

整合服務可降低將開放模型變成客製化營運服務的工作量。底層程式碼免費，顧客仍可能為部署及營運付費。但採用、付費轉換、顧客經濟效益與供應商獲利是不同指標，不能從GitHub星數或公司標誌推論。[3]

更廣的SGLang聯盟是另一協作形式，讓硬體及基礎設施參與者進入共同技術論壇，維護者仍保留技術決策權。經濟邏輯是改善各家公司共用的基礎，不代表共享營收、所有權或顧客資料。[3]

循環圖呈現Miles文件的關係。正式發布被畫成獨立評估決策，不是每次訓練更新的自動後果。[3]

代理讓依賴更強：一項任務可能讀檔、產生程式碼、執行測試、解讀錯誤再重試。每次生成消耗推論，外部工具帶來不確定等待。總實驗亦取決於環境、獎勵計算、訓練及同步；只改善一項，可能暴露另一瓶頸。[3]

6.2 Miles——學習循環需要營運紀錄



圖51｜標示Zhichen Zeng姓名的Miles場次標題。作者會議攝影。[11]

Zhichen Zeng聚焦正式環境等級的後訓練。系統須協調生成軌跡（Trajectories）、獎勵、模型更新與故障恢復，並保存有效學習計算所需資訊。

他列出軌跡生成與訓練可能不一致的來源：詞元處理、專家路由、稀疏注意力、運算核心及非同步陳舊性（Asynchronous Staleness）。這些影響產生嘗試的策略與正在訓練策略間的關係。[11]

擴大RL不只是生成更多嘗試。正式系統需要一致紀錄、版本管理與資料檢查；更快生成只有在學習仍有效、獨立評估改善時才有用。

代理多回合與環境互動，完整歷史及獎勵進入版本化非同步緩衝區（Buffer）。訓練器取批次、計算更新並發布新權重版本。版本標籤記錄哪個模型產生哪批資料。投影片列Megatron-LM、FSDP2及TorchTitan等訓練選擇，與多種權重回傳方式。[11]

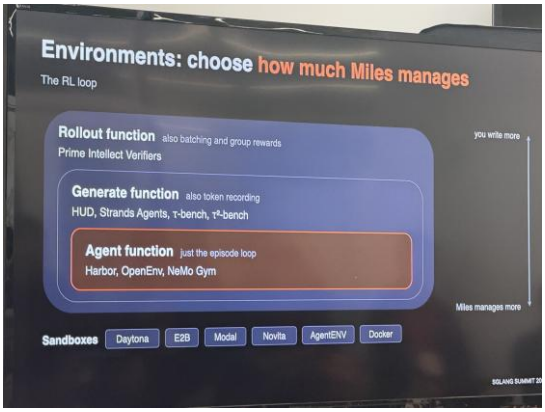


圖52｜Miles後訓練環境。作者會議攝影。[11]

程式練習中，代理修改程式、執行測試、讀錯誤，再修改；通過測試產生獎勵。訓練器用大量此類歷史調整模型。單次成功是訓練材料，本身不證明一般程式能力。[11]

三種整合層級：rollout函式提供較多全流程控制；generate函式包含詞元紀錄責任；agent函式聚焦回合循環，Miles承擔更多周邊工作。[11]

巢狀方框表示責任邊界。外層使用者處理較多事項，包括批次及群組獎勵；內層主要編寫代理回合循環。沙箱提供執行工具的受控電腦；環境定義任務及回饋，沙箱提供執行空間。[11]

工作階段伺服器（Session Server）位於代理執行框架（Agent Harness）與SGLang間。框架管理訊息與工具；伺服器記錄模型精確生成詞元，組成供訓練器使用的歷史。[11]

6.3 異質強化學習（Heterogeneous RL）——協調裝置與策略

Yağız Çalık介紹跨GPU類型與供應商的RL。演講報告48張GPU、四種裝置、兩家供應商的配置，將訓練與軌跡生成分離。[22]

吞吐量與訓練器閒置時間改善屬特定工程配置。小型配接器（Adapter）更新可能比完整大型基礎模型更易分發，但仍需相容性與數值一致性。軌跡吞吐量提升本身不能證明模型能力改善。

軟體將工作安排至適當裝置，可提高異質容量的實用性，增加彈性並減少閒置資源；成本比較仍須納入整合、通訊與可靠性。



圖53｜「在一種晶片訓練，在任何晶片生成軌跡」場次標題。作者會議攝影。[22]

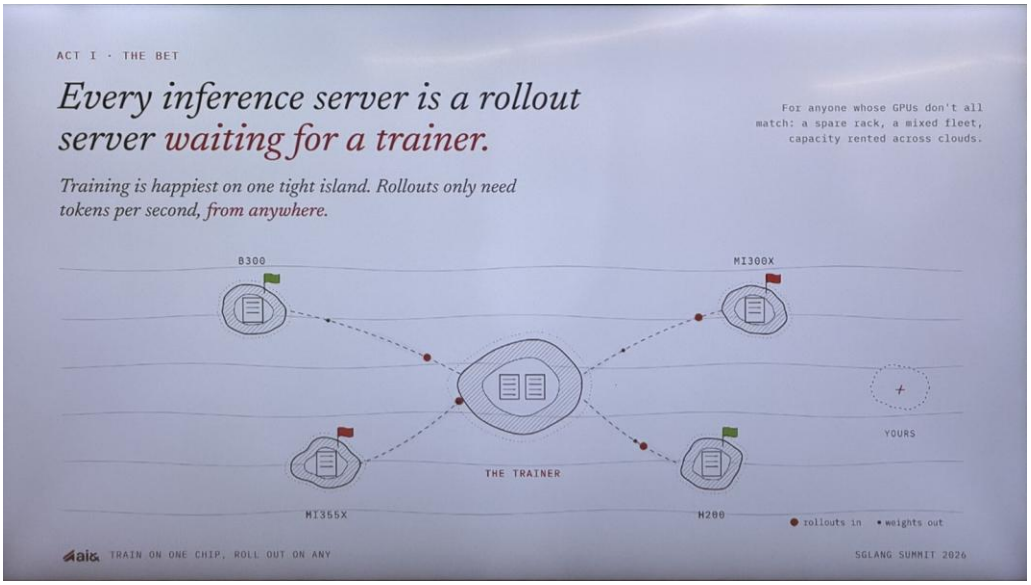


圖54｜把每個推論伺服器當成軌跡生成伺服器。作者會議攝影。[22]

中央島是協調的訓練系統；周邊島在B300、H200、MI355X及MI300X執行模型副本，利用有相容引擎之處的可用容量。「Yours」是邀請整合另一相容部署，不是自動支援的承諾。[22]

軌跡資料回流訓練器，更新設定向外傳播。分散訓練器內GPU頻繁通訊；獨立生成副本可較自主地嘗試不同題目，但每個副本仍可能需要多張緊密相連GPU。[22]

「一個訓練器」不等於一張GPU。節點（Node）是伺服器或運算單元；引擎是服務程序或部署；GPU是實體加速器。數量不能互換。[22]

講者將輝達容量描述為稀缺，列出AMD Instinct、Google TPU、AWS Trainium、Etched、Tenstorrent及Rebellions等替代方案。這是機會地圖，不是每種裝置的測試結果；只有四種NVIDIA與AMD型號被標為通過此次實驗關卡。[22]

若推論引擎產生RL的大量練習材料，組織可能使用相容的閒置推論容量。可用性、記憶體、數值行為與軟體支援都重要，晶片品牌本身不足判斷。[22]

更正說明：口述中「僅用於推論」的描述過廣，Google TPU與AWS Trainium也支援訓練。稀缺性框架是講者主張，不是即時市場庫存調查。[22]

「契約（Contract）」指技術介面約定：引擎須提供相容數值行為、RL特定功能及受支援更新機制，刻意不以廠商品牌為條件。四個通過關卡標記，僅針對測試配置。[22]

6.4 ApprenticeBench——評估持續進行的工作



圖55｜標示Xiang Deng的ApprenticeBench標題。作者會議攝影。[23]

Xiang Deng探問代理能否在持續相關工作中改善。模擬營建會計（Construction Accounting）包含歷史紀錄、公司指令、指導與變動條件；評估的是工作延續性，不是每答完就重設的獨立問題。[23]

研究與NeoCognition相關，比較不同歷史及回饋存取配置。這是從經驗學習的基準證據，不證明自主會計可在無監督下可靠部署。[23]

職場AI應評估延續性、錯誤恢復及適當使用經驗。分數上升亦可能因後續任務較容易，因此須控制難度及回饋取得。本報告關心的是經驗能否在持續工作中改善組織能力。

代理能學會整個職務嗎？考試高分不保證職場能力。應跨相關任務評估，不每題重設；同時衡量初始能力與成長。[23]

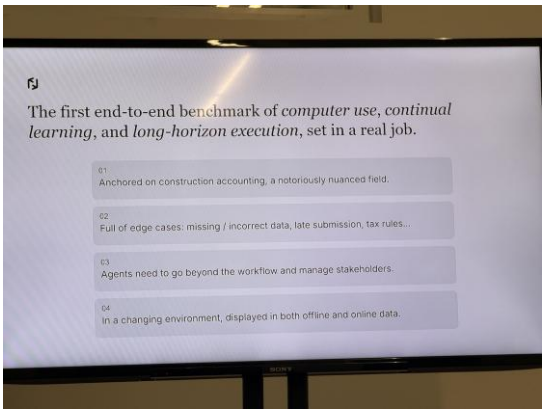


圖56｜ApprenticeBench在工作中學習。作者會議攝影。[23]

一個基準結合電腦操作、持續學習（Continual Learning）與長時程執行（Long-horizon Execution）。缺漏資料、供應商延誤與政策改變會打亂固定流程。營建會計任務含例外與利害關係人，只照腳本不夠。[23]

代理操作商業軟體並比對紀錄。處理一張帳單包含搜尋、檢查與決策：查紀錄、參照先例、輸入適當資訊。一項商業任務可能含數百個介面操作。[23]

影片播放時間、內部任務計時器與成本計數器衡量不同量，評估示範時須分開解讀。[23]

先提供歷史範例，再進入直接及較少介入的指導。新員工需要公司特定知識，因此提供手冊、Odoo教學、已入帳帳單與已完成工作。學習條件也是評估的一部分。[23]

投影片標示5至7月的35項任務，後續結果使用100項。模擬業務月份不代表實際運行了一年。[23]

觀察、決策、點擊或輸入，再確認結果。每個畫面增加資訊，每個動作都可能失敗。代理須反覆解讀介面並保留相關上下文；良好推理必須經得起漫長的小操作鏈。[23]

應用程式介面（API）是接收結構化請求的軟體入口；圖形使用者介面（GUI）是人操作的畫面與按鈕。一次API呼叫不能普遍取代整套流程。[23]

6.5 Axiom Math——將生成與形式驗證結合



圖57｜Tobias Gessler與Rishi Malhotra介紹Axiom Math。作者會議攝影。[24]

演講連結數學生成人工智慧、執行、檢索與學習。AXILE支援證明執行及重用；Atlas處理數學搜尋；Snorlax支援訓練基礎設施。這些共同說明數學公司為何需要大量推論工程。

有價值的單位是通過檢查的數學產物，或通往它的有效步驟。模型可提出很多候選，形式化環境則判定是否滿足編碼後命題。此類回饋不同於主觀文字偏好。

案例展示推論如何配合嚴格否決條件進行搜尋。剩餘邊界在規格（Specification）：在特定定義與公理下有效的證明，不保證原始人類問題被正確形式化。募資、儲存庫規模及解題成就，均須與顧客營收或商業部署分開評估。



圖58｜Axiom Math訓練與證明基礎設施。作者會議攝影。[24]

代理需要既有知識、回饋與訓練流程，否則反覆重建基礎或累積錯誤。[24]

可將三者想成證明工作坊、圖書館搜尋及訓練協調者，模型使用橫跨系統的工具。[24]

自動擴縮（Autoscaling）管理資源；快取產物共享先前建置；持續工作階段重用精化（Elaboration）狀態；工具提供目標、診斷與最終驗證。精化把原始表達式解析成明確結構。[24]

重用可減少浪費，但演講未用對照基準量化節省。快取有效性須跟隨版本與相依項目。[24]

檢索命題、假設、來源與相依關係。Dade等距映射（Isometry）案例顯示連結結果；正數定理不能自動支持所有數的主張。[24]

只有檢索相關且假設吻合，搜尋加速才有意義；未提供精確延遲基準。[24]

未通過檢查可指出剩餘目標或無效步驟，代理再修改或搜尋。最終驗證須保留原命題及接受的假設。[24]

驅動程序分派任務至軌跡生成節點；節點管理沙箱代理工作階段、呼叫AXILE及Atlas，並透過路由器與SGLang取樣器請求輸出。完成軌跡回傳訓練。[24]

SGLang生成候選，Lean相關驗證檢查證明。確切優化器、獎勵權重、GPU數與成本未揭露。[24]

示例：證明提款函式在有效輸入下不會造成負餘額。規格須納入真實規則，包括相關手續費或並行作業。[24]

驗證涵蓋假設下指定性質；部署仍需檢查形式規格之外的風險。[24]

Axiom Math清楚展示有價值的推論循環：模型提出數學工作，檢索提供既有知識，執行環境檢查形式證明步驟，回饋引導後續嘗試及學習。結果不只是一段長答案，而是經明確驗證程序檢驗的候選產物。[24]

驗證改變反覆嘗試的經濟效益。若能可靠拒絕失敗候選，額外推論可用於搜尋，而不用人工逐句檢查。但驗證器只檢查定義及假設下的形式命題，不保證命題符合人類原意，或軟體規格涵蓋所有真實需求。

結果的完成時間改變意義。Axiom區分競賽時限內解出的八道Putnam題目，及之後解出的四道；最終解完12題不同於比賽期間全解。形式化既有數學亦不同於發現新定理，程式行數也不衡量數學深度。[24]

Axiom的策略意義是結合生成與可查核回饋。其他產業可問：能否建立夠強的驗證器或結果衡量，讓反覆推論有用？顧客智慧的衡量可能是經校準預測或對照行銷結果，而非形式證明。

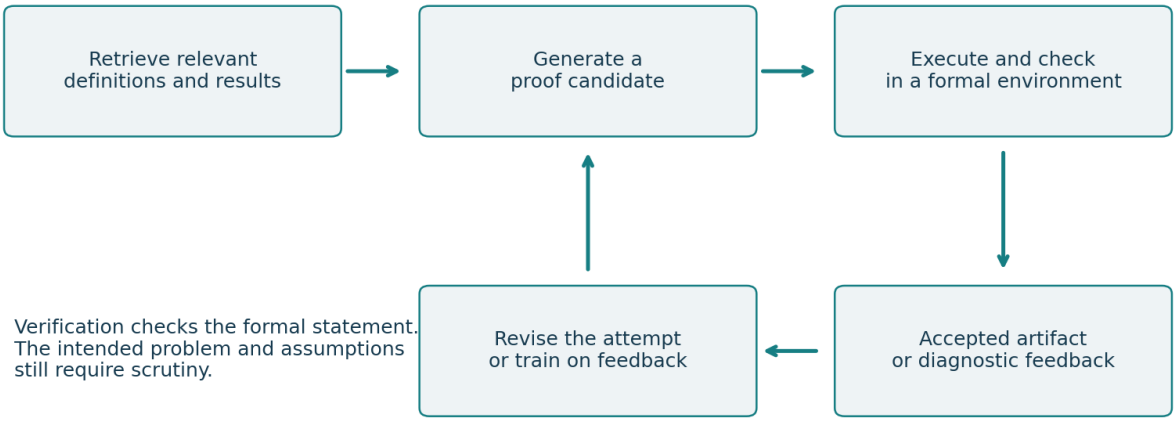


圖59｜生成、形式驗證（Formal Verification）與回饋。GII解說圖；假設見正文。

形式檢查器拒絕無效候選；檢索提供相關命題與相依關係；生成器提出證明步驟或候選，診斷回饋支持再嘗試；訓練用累積經驗改變後續行為。

商業教訓不是每個問題都有數學證明，而是可靠回饋改變反覆推論的效用。程式碼測試套件提供部分證據，顧客鎖定的對照實驗衡量結果。評估器的強度與涵蓋範圍，決定可下何種結論。

通過檢查的產物仍有範圍，須理解其定義、假設與用途。我未獨立重建會議討論的大型形式化專案；完成主張仍歸於Axiom。[24]

6.6 Axiom募資與商業驗證

Axiom公開資料列Carina Hong為創辦人兼執行長，Shubho Sengupta為技術長及創始合夥人，Ken Ono為創始數學家。角色不同，公開措辭差異不能確定法律上的共同創辦人地位或股權。峰會由Tobias Gessler與Rishi Malhotra介紹技術。[24]

有日期的募資揭露提供商業背景。募資額是取得的融資，不是目前剩餘現金、顧客營收或已證明的產品市場契合度（Product-market Fit）。不同輪估值不能相加。[24]

表12｜Axiom已揭露募資輪次

紀錄輪次	募資金額	估值處理方式
2025年10月種子輪（Seed Round）	6,400萬美元	3億美元；估值基礎未指明
2026年3月A輪（Series A）	2億美元	投後估值（Post-money）16億美元以上
已揭露合計	2.64億美元	不同輪估值不可相加

Axiom列B Capital為種子輪領投、Menlo為A輪領投，已揭露募資共2.64億美元。公開個人檔案數與公司規模區間不能確定薪資名冊人數，因此此處不主張精確員工數。[24]

商業評估應從研究產物走向顧客問題。可行路徑可能是將範圍明確的驗證任務整合到開發流程，衡量錯誤或審查工作減少。這是作者分析，不是已確認產品時程或營收預測。下一步需要可重現顧客部署、明確規格及與現行流程的經濟比較。[24]

6.7 Apodex——調查共識背後的證據



圖60｜Simon Shaolei Du介紹Apodex。作者會議攝影。[25]

演講連結多代理研究（Multi-agent Research）、證據查核與未來事件預測。多團隊可以調查同一問題，但只有證據與解讀可靠，共識才有用。

足球案例中，多數代理團隊受到誤導性解讀影響，彙整器（Aggregator）選擇另一答案。案例說明須檢查證據路徑及任務定義，而非只數票；不代表少數答案通常正確。[25]

代理協調（Agent Orchestration）在多元調查、保留來源軌跡及檢查反證時增加價值。反覆使用同一薄弱來源可能產生相關錯誤。結尾Apodex 2.0陳述屬路線圖公告，不代表模型已交付。



圖61｜把推理視為團隊運動。作者會議攝影。[25]

如同新聞編輯室：記者蒐證、分析師解讀、查核者稽核、編輯整合；非同步團隊準備好時各自回傳結果。[25]

最多150個子代理及超過15,000步描述規模，不保證品質，也不是每個任務的必要條件。驗證者也可能失敗。[25]

節點保存發現與暫定主張；邊線表示支持或矛盾。三個網站重複同一來源，不是三項獨立確認。[25]

投影片主張兩項基準分別提升14.8與18.4分；判斷效率仍需基準條件與額外成本。[25]

八條軌跡中五條讀到最可能比數3–0後，選擇三球；但任務其實問總進球數區間。[25]

投影片計算：P（3球）＝P（超過2.5球）−P（超過3.5球）＝85%−63%＝22%；P（5球以上）＝P（超過4.5球）＝45%。六區間機率為2%、4%、9%、22%、18%、45%。[25]

45%不到半數，卻是最大的單一區間。由原始賠率重建機率，須檢查莊家利潤與時間戳記。[25]

彙整器違反五比三的多數，選5球以上；投影片報告德國終場7–1。[25]

驗證器重新解讀任務，使用更相關的總進球分布，查核支持情境，而不只是計算答案數。[25]

案例教訓是重新衡量證據比單純投票更有用，不是推翻多數永遠有效的定理。此處賽果仍是講者報告的結果。[25]

預測執行框架要求每條軌跡質疑初始候選，並提出最多兩個替代方案。[25]

如同準備辯論，找出最強反對理由。每條路徑回傳一至三個候選與信心程度；彙整根據證據包推理。[25]

反思（Reflection）只有改變證據與推理才有用。宣稱的信心程度須跨大量事件檢驗校準。[25]

6.8 Krea——獎勵必須保留預期圖像



圖62｜Sangwu Lee介紹Krea。作者會議攝影。[26]

Krea說明圖像生成與編輯的強化學習。難題是評估同時涉及遵循指令、視覺品質、文字呈現及其他任務要求的輸出。

狹窄獎勵可能鼓勵不良捷徑。文字正確不保證構圖好；懲罰某視覺行為可能矯枉過正而傷害其他目標。Krea例子讓獎勵設計成為具有可見後果的營運問題。[26]

評估須代表顧客真正任務。更多候選提供改善機會，也提供優化錯目標的機會。餐飲零售亦然：點擊率、速度或兌換率改善，整體顧客或商業成果卻可能惡化。

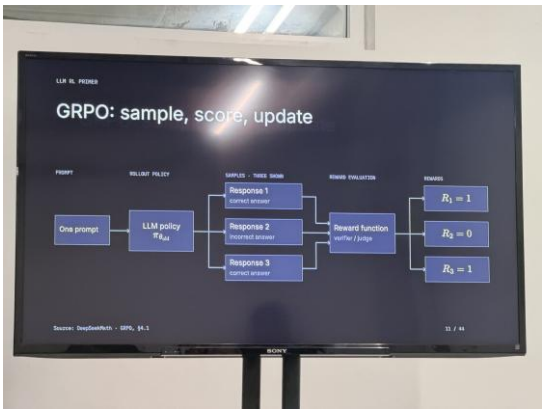


圖63｜Krea強化學習更新。作者會議攝影。[26]

群組相對策略優化（Group Relative Policy Optimization, GRPO）比較群組內獎勵；優勢值（Advantage）表示相對表現。圖示呈現評分，不是完整更新方程式。[26]

雜訊轉為清晰圖像，圖像評分後，再加入新雜訊供訓練。[26]

DiffusionNFT指擴散負向感知微調（Diffusion Negative-aware Fine-Tuning）。學習目標使用獎勵資訊與流匹配（Flow Matching），不必取得完整取樣軌跡。[26]

偏好（Preference）與遵循指令（Instruction Compliance）須分開。即使編輯完全符合要求，觀者仍可能不喜歡風格。[26]

評判器（Judge）若誤讀圖像，獎勵可能懲罰正確編輯，或批准錯誤編輯。[26]

比較預期文字與辨識字元，人工檢視不確定案例。此處雜訊也包含評分錯誤，不只生成時的隨機雜訊。[26]

文字內容與外觀應分開評分，並保留標題與輔助文字的層級。[26]

除了文字辨識器，另加字體排印評判器（Typography Judge），檢查深度、風格及與紙面整合，而不只辨識字元。[26]

代理指標（Proxy）的完美分數，可與不想要的編輯並存；這就是獎勵利用（Reward Exploitation）的例子。[26]

6.9 MiniMax——協調工具擴大輸出形式



圖64｜Open Stage上的Ryan Lee。作者會議攝影。[27]

Ryan Lee探討從語言模型協調走向更豐富多模態（Multimodal）系統。模型可呼叫不同圖像、音訊或程式碼工具；端到端架構追求表示與輸出間更緊密協調。

MiniMax將技術方向連結到中國及國際市場的開發者服務與消費者應用。英文可用性本身不能量化美國營收或證明當地企業採用；地理市場經濟效益需要獨立揭露及定義。[27]

多模態能力符合重複工作流程、且顧客願付費時，才成為生意。服務成本、延遲、品質與留存須共同評估。本綜合報告使用演講技術及策略分析；詳細財務留在有日期的首次覆蓋報告，不當成此次新核實成果。

投影片對照機器文字說明與視覺示範。模態（Modality）是文字、圖像、音訊或影片等資訊形式；空間關係與時序可能更適合視覺表達。同一概念經不同通道呈現，應選擇最有助任務的形式；表達豐富不代表事實正確。[27]

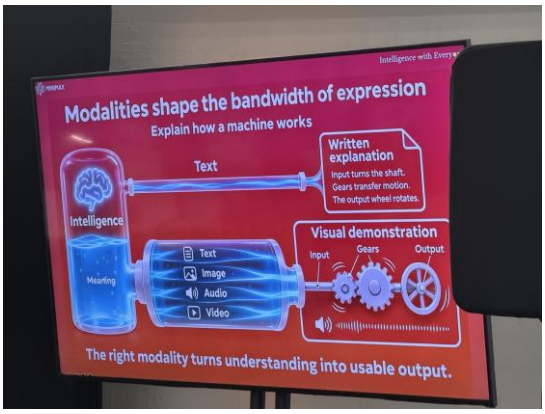


圖65｜模態與表達頻寬（Bandwidth of Expression）。作者會議攝影。[27]

示例：義式咖啡機清潔課可結合正確閥門特寫、旁白與步驟時間。生成影片仍須對照真實機器及核准流程。投影片頻寬比喻指溝通能力，不是實測網路速度。[27]

大型語言模型（LLM）代理分流至模型呼叫與程式碼執行。語言模型不親自渲染每張圖、聲音或遊戲，而是規劃、請求專業生成器，或執行HTML、Blender及遊戲引擎程式碼，再檢查、反覆改進。箭頭匯聚多媒體輸出；整體可靠性比協調模型的流暢度更重要。[27]

示例：餐廳開幕活動需要文案、美術、影片及網站。不同工具可能在色彩、角色外觀或語氣上不一致；審查與重製增加每件驗收活動的有效成本。投影片呈現系統架構，不是正式環境節省實測。[27]

五種交接問題對應聯合表示、串流傳輸、共享上下文、跨模態學習及聯合優化。分離階段可能遺失語氣、增加等待、重解意圖、孤立知識或傳播錯誤；共同訓練的表示可能保留更多關係。端到端是設計選擇，不保證每次部署都更快、更便宜。[27]

這些案例中，推論成為更大改善流程的投入。回饋可信、成本能由較佳驗收結果支持時，更多嘗試才有價值。商業應用因此從生成量，轉向評估器品質與通過評估後的成果效用。

7 經濟效益在哪裡成立

技術改善改變賺錢機會，卻不決定誰能保留利潤。運算設備高度使用，利潤率仍可能壓縮；服務平台簡化部署，卻可能承擔大量營運工作。本章探問容量、協調與差異化服務何處可保留價值，以及哪些假設會推翻結論。

7.1 瓶頸會移動，利潤也會移動

SF Compute將價值獲取（Value Capture）視為有條件的結果。情境分析探討模型能力、運算供給與需求如何演變，不假設AI成長讓每層等比例受益。[28]

雲端賣出更多GPU小時，每小時利潤率卻可能下降；模型商服務更多詞元，價格卻降低；應用對使用者更有價值，基礎設施供應商卻面臨競爭。數量、營收與利潤成長透過價格及成本相連，不是同一結果。

推論供應商的簡化營收為付費使用量×實現價格（Realized Price）。成本含硬體或容量承諾、電力、網路、軟體營運、支援與閒置。會計因業務而異，重點是技術利用率不同於付費利用率（Paid Utilization）：機器忙於重試、內部測試或折價工作，未必支持預期經濟效益。

買方對應指標是：在所需品質及時限下，每個有效結果的成本。即使名目單價較高，若降低失敗、工程工作或部署時間，供應商仍可能有吸引力；反之，快速示範可能很難以買方規模低成本重現。

持續價值獲取取決於哪項限制仍稀缺，以及誰能經濟地解決。可能是晶片供給、電力就緒容量、模型能力、可靠服務、顧客接觸能力或已驗證流程整合。會議資料支持檢視各方案，不證明單一永久贏家。

7.2 SF Compute——價值獲取取決於情境

Ethan Anderson以運算、模型及需求的替代假設，組織產業經濟分析。貢獻是條件式架構：不同組合改變稀缺性及利潤留存位置。

情境矩陣部分格子被遮住。我保留條件邏輯，不重建缺字，也不把每個可見格子當預測。重點是需求、模型與運算假設改變時，答案如何變。[28]

公司可以保有業務，卻失去類似軟體的利潤率。需求增加不保證每個供應商獲更多利潤。分析須區分營收成長、定價能力、成本結構、差異化與資本密集度（Capital Intensity），此架構有助連結會議技術與投資、基礎設施研究。



圖66｜Ethan Anderson解說價值獲取（Value Capture）。作者會議攝影。[28]

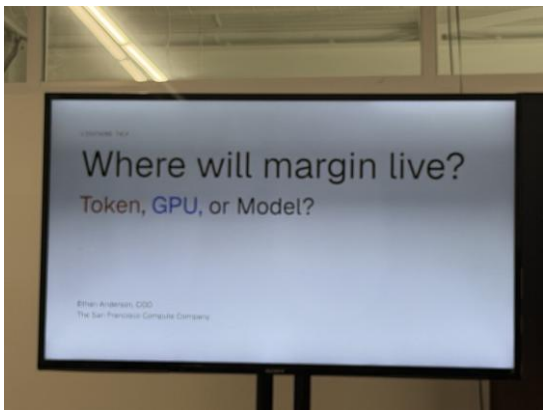


圖67｜利潤會留在哪裡——價值獲取矩陣。作者會議攝影。[28]

三層可能獲取價值：GPU容量、模型及詞元服務。此處詞元（Token）是文字單位，不是加密貨幣。[28]

產業高速成長不保證每個參與者有理想利潤，營收可能只是經公司流向供應商。[28]

應問顧客不容易替代哪種能力，以及提供成本多少。標題提出問題，而非直接選贏家。[28]

晶片類型、叢集（Cluster）大小、位置、付款條件與可用性都影響價格；單一指數可能掩蓋差異。[28]

長租換得低每小時等值價格，不等同彈性逐時計費。同樣，1、2、4、8的成長不同於1、2、3、4；任一比較都不能證明未來價格。[28]

現價描述當前條件，長期利潤預測須明列供給、需求及競爭優勢假設。[28]

示例：需求由100增至150，容量由100增至120，兩者都增長，稀缺性卻上升。交換成長率後，供給就相對充裕。[28]

公司可正確預測AI成長，卻建錯基礎設施數量、世代或位置。此軸比較成長率，不只是成長對衰退。[28]

第二軸問開源在AI總價值中占比變大或變小。[28]

可取得且夠好的模型可能經濟地完成一般任務；獨有前沿能力可能開啟極高價值任務。[28]

示例：開放模型在10美元市場創造5美元，之後在100美元市場創造20美元，絕對貢獻增為四倍，占比卻由50%降至20%。[28]

7.3 Hyperbolic——讓承諾容量對應有用工作

運算網格把單一叢集討論擴展至供應商間協調，擬連結需要彈性運算的顧客與持有可能閒置容量的供應商。

難題在異質性（Heterogeneity）。架構、記憶體、精度、網路、軟體與地理位置影響負載適配。交易平台必須配對這些特徵，不能把所有加速器小時當成相同商品。

更佳協調可減少部分閒置、提高彈性，但不能保證轉售需求、維持所有服務目標或消除合約限制。容量轉售範例是經濟情境，只有相容買方以足以涵蓋成本的價格交易，才成為商業成果。



圖68｜Jasper Yue Zhang解說運算網格（Compute Grid）。作者會議攝影。[29]

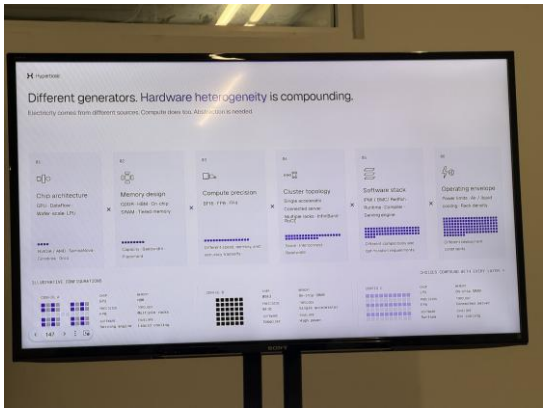


圖69｜運算網格架構。作者會議攝影。[29]

六個不同面向：晶片架構、記憶體、數值精度、網路拓樸（Network Topology）、軟體堆疊，以及電力或冷卻限制。[29]

依能力配對負載。容量控制能放什麼，頻寬控制資料移動；低精度可省資源但需準確性檢查；網路拓樸控制通訊。[29]

要求八張相容GPU使用六小時，需要資源搜尋、軟體檢查、存取、監測及計費。這是功能解說，不證明全部功能已部署。[29]

可轉售安靜時段或未用承諾，同時保護下一項排定工作。升級可能結合轉售舊硬體承諾與租用新容量。租戶隔離（Tenant Isolation）、權限與計量不可少。[29]

必須在適當時間找到買方。相關高峰、舊硬體降價與合約限制，都可能限制成本回收。[29]

示範預訂1,000張B300 GPU七天，每承諾GPU小時4.50美元：1,000×168＝168,000 GPU小時，原成本756,000美元。假設轉售回收268,800美元，淨成本降至487,200美元，即每承諾GPU小時2.90美元，回收原帳單35.6%。這是情境假設，不是報價或顧客已實現節省。[29]

分母是承諾小時，不是自用小時。實際節省取決於售價、費用、時點與需求，並無保證。[29]

結語中Jasper Zhang提出跨雲優化：顧客指定負載、預算及期限，平台選資源。他將推論描述為既有業務，沙箱與訓練服務則在演講當時逐步推出。這是平台方向及自述狀態，不是獨立可用性測試。[29]

例如夜間處理10,000張圖片，須檢查相容軟體、記憶體、預期速度、資料傳輸及總成本。期限較寬鬆可選較便宜機器。這是解說案例，不是已測試產品方案。[29]

顧客節省、供應商營收與利用率可能衝突。較高價格利於賣方、卻不利買方；遷移可增加硬體可用性，也可能增加傳輸成本。優化需要限制及取捨，不能承諾所有目標同時最大化。[29]

7.4 SkyPilot——協調決定有效利用



圖70｜Zhanghao Wu介紹SkyPilot。作者會議攝影。[30]

演講將RL描述為不同工作間的協調問題。軌跡生成、訓練及支援服務須在適當時間取得資源。有些工作要一組資源同時啟動，部分配置反而浪費。

群組排程（Gang Scheduling）、配置與資源協調可改善容量使用。投影片利用率是百分比，不是加速倍數；排程更有效，本身不證明模型品質提升。[30]

工作負載與可用容量間的軟體層可有重大經濟價值。應以有效任務完成、閒置減少及營運簡化衡量，並納入工程與服務成本。

Qwen3-4B控制四推進器模擬載具。40億參數是模型學得的數值設定，不是工程師寫的40億條規則。[30]

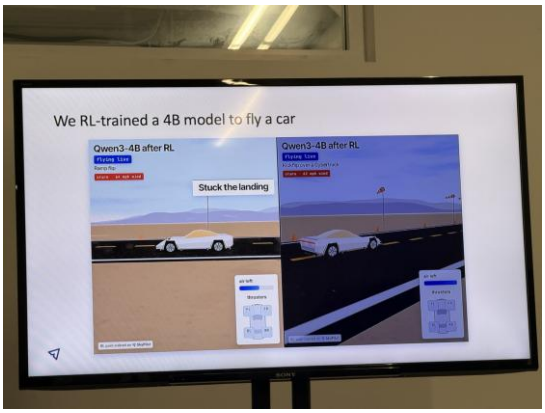


圖71｜SkyPilot中的模擬控制負載。作者會議攝影。[30]

升力、旋轉、風與降落需協調；升力更多不一定好，載具還須安全返回降落區。[30]

控制間隔0.2秒，即1÷0.2＝每秒五次決策。受力不均會旋轉，如只推書的一邊而非兩側平衡抬起。實際物理由模擬器決定。[30]

動作必須現在有效，也留下讓下一動作能成功的狀態。[30]

圖表區分決策品質與回應延遲。暫停的模擬器問動作好不好；持續移動的模擬器問動作是否好且及時。延遲是一個回應的時間，吞吐量是單位時間總工作。[30]

模型整體可服務很多請求，卻仍可能對即時控制器太慢；晚到答案可能回應已不存在的狀態。[30]

公司報告storm-kickflip成功率：未訓練0%、模仿學習（Imitation Learning）後84%、模仿加RL後90%，訓練後回應約0.09秒。84至90是增加六個百分點；所選訓練及模擬條件限制外推。[30]

針對性後訓練可能有價值，但結果不證明單靠RL永遠勝出。13美元訓練數字不是研究或原始預訓練總成本。[30]

推論生成動作，環境執行，觀察與獎勵記錄結果，訓練器更新權重，再交回推論。[30]

7.5 容量、利用率與邊際貢獻是不同指標

Hyperbolic願景在於配對異質容量與需求。硬體的記憶體、互連、軟體支援、地點與電力不同，因此GPU小時不能完全互換。顧客買的是可用配置，不是抽象數量。[29]

未用承諾容量有時可轉售或改派，但須有相容需求，且商業技術安排允許。轉售案例是情境，不是保證營收；錯時間、錯區域或錯網路的容量可能很難變現。

SkyPilot聚焦跨資源協調RL負載。有些階段須同時開始，其他可獨立。群組排程是把必要資源作為整組配置；緊密耦合工作只啟動部分、其餘等待，可能浪費容量。[30]

這些觀點連結財務與工程。長期承諾可降低單價卻暴露閒置風險；彈性容量降低承諾風險，卻可能更貴或更難取得。協調軟體改善使用，不會創造顧客或消除所有相容限制。

研究須問兩面：系統能排程多少有效工作？其中多少具經濟價值或有人付費？嚴謹基礎設施模型須包含兩者；未定義利用率不足支持投資結論。

7.6 讓營收假設清楚可見

純示例：100個加速器在30天月份各有720小時。付費利用率60%時，出售43,200加速器小時；每小時2美元，營收86,400美元。利用率80%、同價則為115,200美元。這是算術情境，不是市價或硬體採購建議。

敏感度表說明價格與付費利用率同樣重要。需求不足時名目高租金無用；機器滿載但大幅折價，也可能不敷總成本。分析應區分承諾容量成本與變動費用，適當納入融資或折舊。

完整投資模型還需硬體購入或租賃條件、使用年限、殘值、耗電、電價、冷卻、網路、人員、服務抵扣與顧客集中度；須測試爬坡期及停機，不能假設立即穩態。此例只隔離利用率與營收，不是特定設施的經稽核成本模型。

餐飲買方通常透過供應商價格與可用性間接感受這些經濟因素；對GII資料中心研究則是核心。推論負載創造需求，也決定服務輪廓：延遲敏感服務、批次分析與RL生成，配置及排程需求各異。

敏感度表（Sensitivity Table）的目的在揭露假設，讓讀者替換數值後檢查結論是否仍成立。

表13｜假設每月總營收：100個加速器×720小時

付費利用率（Paid Utilization）	每小時1.50美元	每小時2.00美元	每小時2.50美元
40%	$43,200	$57,600	$72,000
60%	$64,800	$86,400	$108,000
80%	$86,400	$115,200	$144,000

7.7 信任與可靠性屬於商業服務內容

Phala機密推論保護運算中的資料。驗證證明（Attestation）在明列信任模型下，提供環境與軟體身分證據；這是執行證據，模型答案是否正確仍是另一問題。[15]

Palo Alto Networks處理另一問題：安全服務在嚴格延遲目標內執行模型。實測配置重要，達標同時客戶端增加不自動等於總吞吐量同比增加；服務整體請求數也不是單GPU基準。[21]

顧客智慧中，機密性決定資料在哪處理、誰能接觸；準確性決定擷取主題與預測是否可靠；安全與營運政策決定允許哪些行動。每項需求都需自己的控制及證據。

此區分亦改善商業報導。「安全AI」若無威脅模型（Threat Model）過於空泛；「準確AI」需任務與評估集；「快速AI」需負載與延遲分布。廠商可能一面進步、一面付出代價。

信任是交付服務的一部分，影響基礎設施選擇、採購審查及部署應用範圍，應與記憶體、運算與價格同樣具體分析。

營運上應一起辨認買方、任務、約束資源及收費模式。容量賣方、軟體商與應用業者都可受益於推論需求成長，卻面對不同經濟條件；有效產業評估應明確呈現差異。

8 在30萬名顧客的企業中理解10萬份顧客資料

我對推論的興趣最終回到商業問題。假設餐飲公司服務30萬顧客，資料庫有10萬份可識別檔案，如何了解他們是誰、需要什麼、未來可能貢獻多少，以及哪種接觸有用？以下數字是假設研究設計，不是已分析資料集的描述。

8.1 核對顧客母體（Customer Population）

假設公司在明確12個月期間估計有30萬名不重複顧客，第一方資料庫（First-party Database）含10萬份可識別顧客檔案。這是全章示範情境，不是觀察到的公司資料。

首先確認兩個母體是否可比。「顧客」可能指個人、帳號、家戶、支付識別代碼或外送平台使用者。重複帳號可膨脹資料庫，共用付款方式可合併多人，未識別訪客可能只出現在交易總額。精準分析從定義與核對開始。

即使10萬檔案皆不重複，也不等於全可接觸。有些無有效管道、有些拒收、有些過時、有些僅允許特定用途；也不自動代表其他20萬人。忠誠會員與偶爾來客可能有系統性差異。

目標不是替所有人虛構個別檔案，而是理解已知、衡量可估計部分、辨認不確定性，並在限制內改善決策。門市整體模式可協助規劃匿名需求，不必假裝知道每個人的身分。

管理問題很具體：哪些需求反覆出現？誰可能再訪？哪些關係扣成本後有貢獻？誰可能因介入改變行為？適當管道及時機是什麼？各問題需要不同證據路徑。

表14｜顧客母體核對

母體	示範人數	解讀
期間內不重複顧客總數	300,000	需要定義與估算方法
可識別資料庫檔案	100,000	必須去除重複並核對
未包含於可識別檔案的顧客	200,000	僅在兩母體真正可比時成立
可接觸且符合資格的檔案	未知	不能從資料庫大小推得
符合實驗資格的檔案	未知	取決於管道與實驗設計

8.2 建立顧客證據表（Customer Evidence Table）

有效顧客表是有組織的證據紀錄，可包含穩定內部識別碼、來訪日期、消費、地點、管道、商品、退貨、優惠兌換、同意狀態與回饋。每欄保留來源及更新日，分析結論應可追溯至支持紀錄。

身分解析（Identity Resolution）優先。已驗證帳號等確定式配對（Deterministic Matching），通常比機率猜測易於辯護。必要的不確定配對應保留信心與原紀錄；不能只因同住址或姓氏就合併兩人。錯誤合併污染預測及溝通。

交易表與檔案表用途不同：前者記事件，後者摘要關係。彙總特徵（Features）可包括距上次來訪天數、回溯期來訪次數、平均貢獻、時段與管道組合。特徵須有時間窗口，沒有期間的頻率無法解讀。

缺漏值（Missing Values）須有意義。「無飲食偏好紀錄」不同於「沒有飲食偏好」；「沒有電子郵件」不同於「不想收郵件」。語言模型不能用看似合理的個人屬性補空白。接觸顧客時，不確定推論應視為假說，並提供更正途徑。

專案可從適度資料模型與可重現更新流程開始。早期最高價值結果可能是確認10萬檔案中究竟多少可用於分析及聯繫；即使尚未跑複雜模型，這已是商業發現。

表15｜最小證據資料結構（Evidence Schema）

欄位群組	範例	控制措施
身分（Identity）	內部顧客識別碼（ID）	去除重複與配對來源追蹤
行為（Behavior）	來訪日期與品項	明確觀察期間
經濟效益（Economics）	扣變動成本後的貢獻	與會計資料核對
管道（Channel）	電子郵件 / 應用程式 / 簡訊資格	最新偏好紀錄
文字證據（Text Evidence）	抱怨或評論摘錄	來源與信心程度
模型輸出（Model Output）	預測下期邊際貢獻	版本與評分日期

8.3 選演算法前先理解需求

觀察行為可暗示需求，卻不證明。平日午餐常來的顧客可能重視速度或地點方便，僅交易模式不能判定是哪個。文字回饋、短問卷或良好實驗可補充證據。

有效分析結合多重視角：地理模式說明需求在哪；時段與場合說明何時；商品及管道說明選什麼、如何交易；文字解釋部分原因。整合結果比單一人口統計標籤豐富，尤其尚未真正蒐集人口資料時。

LLM可把回饋分類至受控主題，如等待時間、點單正確性、份量一致性、價值感或款待。輸出須含來源摘錄與不確定標記。文字不足支持時，應回覆「證據不足」，而不是發明原因。

母體摘要須報告分母。已分析抱怨中12%提到等待，不等於全部顧客12%等待過久。留下回饋者是經選擇的群體，須說明分析多少文字、如何抽樣、同一人多次留言是否分開計。

語言模型最有力的初始用途，可能是將分散質性資料轉為一致、可稽核資料集，補充結構化分析；不是取代交易帳冊，也不是把未觀察屬性變成事實。這是推論基礎設施與顧客理解的實際連結。

8.4 以明確期間估算顧客價值

歷史顧客貢獻（Historical Customer Contribution）衡量既有關係依指定會計定義創造多少；預測顧客終身價值（Predicted Customer Lifetime Value, CLV）估未來貢獻。兩者應分列：過去高價值者可能不再回訪，新關係歷史少卻可能潛力大。

可管理的起點是12個月預期貢獻，而不是無邊界「終身」。對每人估未來12個月來訪或購買次數、每次貢獻及關係成本。進階模型可處理行為變化、不確定性與折現；清楚基準優於難以解釋的複雜數字。

假設顧客下一年預期來八次，每次在個別行銷費前貢獻9美元，預期貢獻72美元。若個別溝通費4美元，其他調整前淨估68美元。這是教學示例，不是真實顧客母體預測。

貢獻指標須反映決策。折扣、管道佣金、退款及變動成本影響新增來訪價值。把所有固定成本攤到顧客，可能掩蓋邊際行銷決策；但完整投資評估忽略固定基礎設施成本也錯。應說明指標回答哪個問題。

價值模型應提供不確定性，至少區分穩定估計與資料稀少估計。一訪檔案不應與多年穩定觀察有同等信心。目標是改善優先順序與學習，不是製造假精準。

表16｜12個月邊際貢獻示例

輸入	假設	結果
預期來訪次數	8	預測，不是保證
每次來訪貢獻	$9	預期來訪貢獻72美元
關係維繫成本	$4	淨預期貢獻68美元
不確定性（Uncertainty）	此簡化範例未估計	實際營運模型必須處理

8.5 從過去學習，避免未來資料洩漏

預測未來12個月，應使用較早時點可取得資訊建立歷史範例，特徵摘要前六或12個月，目標記錄後續期間實際結果。先前觀察搭配未來結果，形成監督式學習（Supervised Learning）資料集。

簡單基準可用最近來訪（Recency）、頻率（Frequency）及貢獻分組。較彈性模型可直接預測回訪機率、購買頻率或未來貢獻。選型取決於目標、資料量、缺漏與解釋需求；結構化交易表的最佳數值預測器不一定是語言模型。

時間切分不可少。若特徵誤含結果期來訪，模型就偷看答案，回測（Backtest）可能很好，上線卻失敗。應依時間切分訓練與評估，使用未參與選特徵或調參的期間；資料允許時，跨多個歷史截止日重複檢驗。

除了排序（Ranking），也衡量校準（Calibration）。若被預測30%回訪的人，在可比評估集約30%真的回訪，預測即符合校準。模型可排序不錯，卻系統性高估機率；預算依預期貢獻制定時尤其重要。

最後與簡單營運政策比較。若扣實作成本後仍無法勝過最近來訪 / 頻率基準，複雜度尚未證明價值。初次可採影子模式（Shadow Mode），只產生分數、不改變顧客待遇，同時檢查資料品質與穩定性。

表17｜實用模型開發流程

步驟	交付成果
定義目標	未來90天回訪或下一年貢獻
固定觀察日期	只用該日已可取得的資訊
建立基準（Baseline）	最近來訪 / 頻率 / 貢獻規則
訓練與驗證	時間順序留出樣本及資料洩漏檢查
評估商業效用	校準、排序與決策成本
審慎部署	版本化分數及持續監測

8.6 語言模型推論在哪裡增加價值

語言模型可擷取回饋主題、標準化凌亂文字、檢索營運資訊、用白話解釋模型輸入，以及草擬待審查訊息。各角色須有明確輸出邊界與可測試品質標準。

回饋擷取要求結構化紀錄，含主題、情緒（Sentiment）、支持語句及信心程度，提供允許類別與拒答選項（Abstention）。用人工標註樣本評估，包含短、模糊及多語言評論；簡單案例高準確率不能掩蓋困難群體的系統錯誤。

分析解說應讓LLM取得已驗證數字與定義，不要求憑空計算。確定式查詢或模型算分，語言模型說明結果及限制。證據只說來訪減少時，不能無依據斷言顧客搬家或不喜歡餐點。

聯繫使用受控且已核准優惠與主張。系統可調語氣或摘要利益，但資格、接觸權限、預算與優惠規則須明確控制。敏感或特殊案例可人工審查；監測包含抱怨與取消訂閱，不只點擊。

這直接連結峰會基礎設施主題：大量短分類可用批次服務，互動員工協助需低延遲，共同指令可快取，顧客資料邊界可能需機密或本地受控部署。架構跟隨實際用途，而非一律選最大模型。[12, 15, 1]

8.7 接觸顧客時，別混淆回應與影響

高回應機率不代表促銷創造額外價值。有些忠誠顧客無折扣也會來，發優惠可能降低貢獻，兌換率卻漂亮。因果問題是行為因介入（Intervention）改變多少。

隨機留出對照組（Randomized Holdout）是有效起點。合資格顧客隨機分派至介入或現行政策組，推出前定義觀察窗口及結果，衡量增量貢獻（Incremental Contribution），不只來訪或總銷售；包含折扣、溝通及相關變動成本。

假設20,000合資格顧客隨機分派：10,000收優惠、10,000為對照。相同觀察期內，處理組600人、對照組500人各購買一次，教學例假設無重複購買。購買率6%與5%，差一個百分點，等大群組相差100次購買。這是點估計（Point Estimate），仍需評估不確定性；把600次全歸優惠會高估效果。

增益模型（Uplift Model）估不同顧客的處理效果差異，需適當實驗或其他可辯護的因果資料。一般購買機率模型，不能因用來鎖定顧客就改稱增益模型。

長期可學習哪些需求、管道與優惠，對哪些群體產生增量貢獻。學習目標須保護顧客關係：重複訊息、折扣依賴及流失會抵銷短期收益。此時顧客價值才是決策系統，而非單一分數。

表18｜行銷活動對照示例

指標	處理組（Treatment）	對照組（Control）
合資格顧客	10,000	10,000
購買一次的顧客	600	500
顧客購買率	6%	5%

8.8 計算行銷活動經濟效益

延續假設實驗，每次購買折扣前貢獻12美元，處理組每筆兌換2美元折扣，對照組無折扣。處理組600×（12−2）＝6,000美元，對照組500×12＝6,000美元。簡化假設下，溝通費前購買增加卻未增加貢獻。

若聯繫10,000名處理組顧客成本500美元，觀察到的淨貢獻差為負500美元。案例說明購買率提高仍可能破壞價值；折扣不只給新增顧客，也給了原本就會買的人。

計算刻意省略長期效果、客單變化、容量限制及統計不確定性。真實實驗若影響重大須納入：活動可能建立新習慣、改善後續貢獻，也可能只是把下週來訪提前。觀察與追蹤期間因此重要。

推論的教訓是：只有決策規則經濟上合理，更準預測才有用。準確找出可能買的人，可能放大錯誤促銷；較簡單模型搭配良好實驗，反而帶來更有價值學習。

這為GII連結GPU效率與餐飲成果：基礎設施決定分析多便宜、多快；實驗衡量決定行動是否創造價值，不能把一者當作另一者的證明。

表19｜邊際貢獻推導示例

計算項目	處理組（Treatment）	對照組（Control）
購買次數	600	500
折扣前貢獻	$7,200	$6,000
折扣成本	$1,200	$0
溝通成本	$500	$0
淨貢獻（Net Contribution）	$5,500	$6,000

8.9 讓模型服務設計對應決策

並非每種顧客分析都需不到一秒回應。夜間評分可優化成本並於早晨期限完成；員工在顧客互動時詢問解說，有另一延遲目標；月研究報告可接受比即時推薦更多驗證及更長計算。

10萬檔案不代表10萬個長生成提示。結構化模型可有效逐列評分，文字擷取只處理相關新增或變更紀錄，檢索只提供問題所需證據。重用穩定指令或已驗證中間結果，可在正確性及存取控制下減少重複。

首次部署採託管服務可能合理，因公司購買的是有用分析，不是GPU營運部門。當規模、資料需求、可預期性或特殊能力足以支持營運責任，再評估自架（Self-hosting）；兩者沒有普遍較便宜的一方。

完整成本包括資料準備、模型呼叫、工程、監測、審查、失敗工作及下游重工。詞元費只是其中一項；低單價可能被過長提示、重試或需人工修正的低品質輸出抵銷。

觀察真實負載後應重訪部署決策。當團隊知道提示分布、輸出長度、並行量、品質門檻與期限，PD、快取、記憶體及協調經驗才可採取行動。實測負載是餐飲問題與基礎設施採購間的橋梁。

表20｜工作負載與架構對照

工作負載（Workload）	主要目標	可能的起始設計
夜間數值評分	期限前完成	結構化批次流程
回饋分類	準確性與每筆驗收紀錄成本	批次LLM擷取
主管研究問題	有依據且附來源的回答	檢索搭配已驗證表格
即時員工協助	延遲與安全轉交處理	具監測的互動服務
行銷活動學習	因果增量貢獻估計	實驗與分析流程

8.10 九十天研究流程

第1至30天：建立定義、資料存取與基準。核對30萬顧客估計及10萬檔案，辨認重複與可接觸者，定義貢獻，記錄同意及管道狀態，選一個商業問題。例如：已記錄抱怨後，服務補救訊息是否改善回訪？

第31至60天：建立可重現分析流程。建時間明確的特徵，實作簡單預測基準，以標註範例評估文字分類，產生含不確定性的評分表；採影子模式執行。員工應能說明每個數字來源，辨認系統應拒答的情境。

第61至90天：以留出對照測試一項受控介入。預定目標母體、接觸規則、優惠、結果與觀察期間；檢查門市能否兌現、員工能否答疑、企業能否一致記錄。技術精巧的活動也可能因營運流程不完整而失敗。

交付不只模型，而是從紀錄到決策的可靠證據路徑，含實測結果及下一個問題。若首次實驗無收益，仍可因避免昂貴全面推出、找到更好假說，而取得學習上的成功。

這是作者依峰會系統經驗提出的應用架構，不是講者宣稱已部署的餐廳案例。實際試點須取得組織資料、人力、決策時程及限制，才能設定預算與績效目標。

8.11 完整顧客智慧循環（Customer-intelligence Loop）

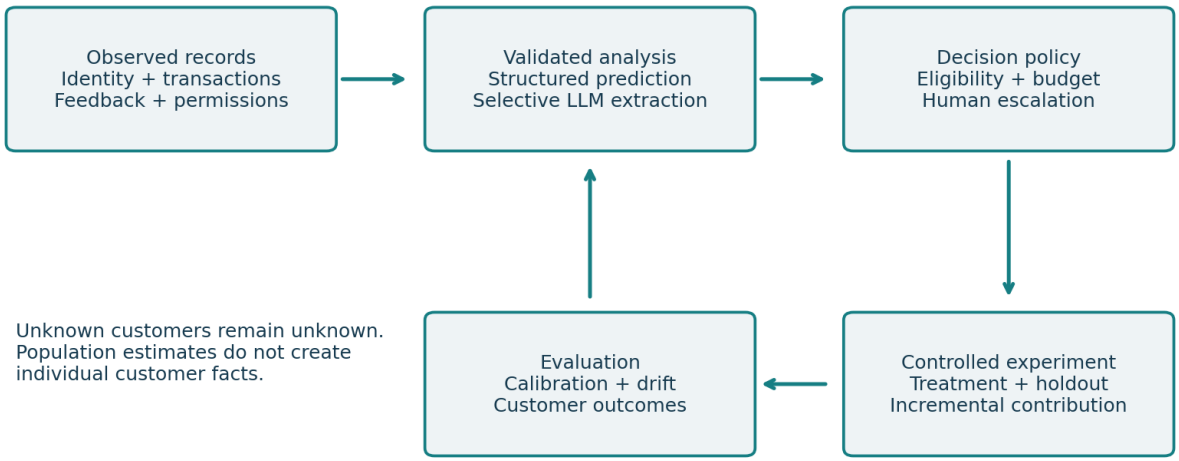


圖72｜餐飲顧客智慧營運循環。GII解說圖；假設見正文。

循環從觀察紀錄開始，以實測結果結束。預測估可能發生什麼；政策決定採哪個合資格行動；實驗衡量行動是否改變貢獻或顧客行為。

回饋回到分析階段，但模型更新不應自動。團隊須檢查結果改變來自顧客行為、資料品質、新優惠、營運干擾，還是模型真正改善；版本化紀錄使調查可行。

母體邊界始終清楚：10萬可識別檔案不會創造其餘顧客的個別知識。更好分析可改善母體估計並引導蒐集，不必假裝不確定性消失。

9 對餐飲、機器人與資料中心的意涵

峰會擴展第一集問題。餐廳實體營運、機器人動作循環、資料中心模型服務容量，都依賴整套系統在當地限制下運作。有效比較是能力如何成為可重複工作，以及哪些證據支持擴大投入。

9.1 餐飲經營者需要改變什麼

第一步是定義可衡量的顧客決策，從可識別群組、清楚結果期間及小規模測試開始，檢查預測或訊息是否改善貢獻。選擇能回答問題的最簡分析方法，再於能產生有用、可查核工作之處加入語言模型。

這與第一集相連：據點承擔營運後果；任務定義學習與驗收；網路決定成功方法能否轉移。局部試點改善結果，仍可能須調整資料蒐集、員工流程及權限，才能跨連鎖體系重現。

9.2 Generalist——推論與實體動作交會

Felix Wang介紹實體提示（Physical Prompting）：簡短示範進入上下文、不更新權重，也能影響機器人行為；此機制不同於為每個新任務微調策略。

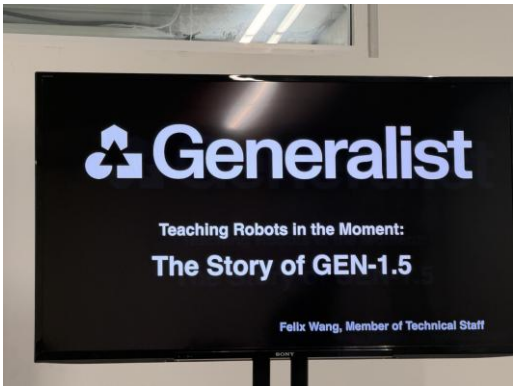


圖73｜Generalist GEN-1.5場次標題。作者會議攝影。[10]

演講結合實驗說明、示範畫面及語言模型發展類比。底層參數固定，暫時上下文仍可改變下一動作；理解此區別是解讀示範的核心。[10]

快速適應可提升彈性，但短示範不證明餐廳、倉庫或零售環境的全面可靠性。任務分布、恢復行為、硬體耐用性及安全邊界仍須衡量。「GPT-3時刻」是講者比喻，不是獨立證實的採用時程。

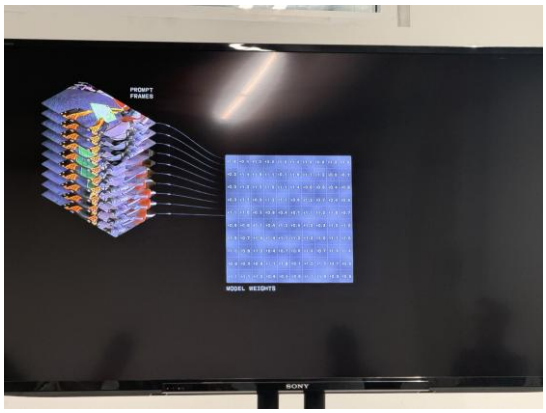


圖74｜實體提示與已學習模型權重。作者會議攝影。[10]

一疊提示畫格連到「模型權重」方塊；數字是解說繪圖，不是公布的網路規格。[10]

輸入與儲存參數的區別很重要。權重不變，機器人仍可因輸入改變而改變動作。[10]

可把權重想成練熟技能，提示畫格是桌上當前範例；上下文暫時引導技能。[10]

實體提示改變提供給模型的資訊；微調（Fine-tuning）改變模型本身，屬不同適應機制。[10]

提示畫格與當前輸入共用上下文窗口（Context Window），輸出形成機器人軌跡。較早動畫畫面的軌跡區是空白，後一畫面才呈現路徑。[10]

機器人須把範例連結到當前場景，物體移動時，照抄精確座標會失敗。[10]

模型依範例與當前觀察預測動作。軌跡（Trajectory）是一段時間的位置或動作序列，須由實體系統執行。[10]

圖解說資訊流，不是所有軟體元件；未指定模型大小、注意力配置、控制迴圈時間或安全機制。[10]

問題是機器人能否利用不到一分鐘示範，且零次梯度更新（Gradient Updates）就適應；此稱上下文學習（In-context Learning）或實體提示。[10]

避免逐任務優化可縮短設定，提供新範例後即可改變行為。[10]

把範例放進模型上下文記憶，而非用它重寫權重；預訓練模型在推論時解讀上下文。[10]

零更新不代表零先前訓練、零運算或訓練語料中零任務經驗，只指此階段的適應方式。[10]

9.3 資料中心容量必須轉為明確服務

硬體照片讓推論具體：密集系統需要供電、散熱、資料移動與維修空間。設施及網路設計決定處理器能否按預期工作。[7, 6]

已安裝百萬瓦（MW）與加速器衡量實體容量；可銷售推論容量還取決於負載、軟體、記憶體、互連、服務目標與可用性。因此電力容量相近的設施可有不同商業輪廓。

採購需有日期的完整配置，涵蓋運算、網路、冷卻、設施工項、支援與部署。第四章照片說明這些為何須一起討論。示範採購預算不是廠商報價。[7, 6]

Uni³Sim模擬運算、記憶體、互連與排程的交互作用；若假設與預測經實測系統校準，可在部署前篩選配置。[20]

對GII而言，投資審核（Underwriting）鏈條從實體容量走向負載層服務，再至營收或內部價值，連結選址、工程與商業盡職調查（Commercial Diligence）。

9.4 我會帶進下一次部署的問題

先問顧客任務：什麼輸出、什麼品質、何時完成、多少使用者？這些要求引導模型、服務配置、硬體及成本。

效能主張須提供模型版本、精度、輸入 / 輸出分布、並行量、快取政策、硬體數與服務目標；記錄失敗及逾時，區分平均、百分位與最佳測試。

商業主張須追蹤改善是否帶來較少工程、更多有效請求、更強留存，或先前不可行的應用。買方需要這條價值連結。

採用主張須區分示範、試用、付費部署與大規模持續使用。募資記為融資，標誌記為參與，路線圖記為未來意圖。

學習系統須辨認可信回饋及訓練循環外的評估。生成輸出成為未來訓練材料時，更多運算既可放大有效搜尋，也可放大系統性錯誤。

這些問題讓技術報導易懂，同時保留專家挑戰結論所需細節。

9.5 下一步研究議程

推論是系統與營運經濟學問題。模型能力依賴排程執行、正確狀態管理、可靠工具，以及能辨識有效輸出的評估。

預填充 / 解碼與顧客活動共享教訓：保留分母。單人串流更快可伴隨總吞吐量不變；購買率更高可伴隨貢獻更低。標題改善與商業結果都須衡量。

硬體與平台案例說明大廠參與原因。共用軟體可擴大競爭者採用，整合、支援與營運效能仍是差異化來源。

Axiom Math、Krea、Apodex、Miles與Generalist把推論延伸到搜尋、驗證、學習、創作與動作。各自需要適當驗收：形式驗證器、獨立評估、對照實驗或實體成功標準。

我下一步GII研究重點，是連結這些機制與營運證據：用真實資料集做顧客價值分析、依負載分析基礎設施經濟效益，以及在真實環境衡量實體AI部署。從餐廳據點開始的問題，到伺服器機架仍適用：整套系統是否以可接受成本交付有用工作？

我帶著經營者對AI可能性的興趣來到峰會，帶著更精確問題離開：有前景的能力進入真實企業前，必須先衡量什麼？路徑穿過顧客證據、模型行為、系統工程與營運經濟學。沿此路徑，GII才能把餐廳現場連結到推論經濟。

附錄A 實用術語表（Working Glossary）

實用術語表

表21｜術語表A——模型服務與硬體

術語（Term）	定義
推論（Inference）	將訓練完成的模型套用於新輸入。
預填充（Prefill）	處理輸入序列，準備生成所需狀態。
解碼（Decode）	利用可用上下文產生後續輸出詞元。
鍵值快取（KV Cache）	供注意力使用的已儲存鍵 / 值狀態，不是已完成答案資料庫。
預填充與解碼分離（PD Disaggregation）	將預填充與解碼放在不同服務資源。
首詞元時間（TTFT）	到首個輸出詞元的時間，包含相關非引擎延遲。
詞元間隔（ITL）	生成詞元之間的時間間隔。
吞吐量（Throughput）	受測系統每單位時間完成的工作。
有效吞吐量（Goodput）	符合明確品質與服務目標的有效完成工作。
批次（Batch）	一起處理的一組工作項目，具體意義依引擎而定。
並行量（Concurrency）	同時進行中的請求或活動數。
服務水準目標（SLO）	定義明確的服務目標，不同於合約型SLA。
高頻寬記憶體（HBM）	位於加速器附近的高頻寬記憶體。
記憶體分層（Memory Tiering）	跨不同容量與速度的儲存層管理狀態。
混合專家（MoE）	選擇性啟用專家元件的模型架構。
運算核心（Kernel）	實作特定運算操作的裝置程式。

實用術語表（續）

表22｜術語表B——學習、信任與顧客分析

術語（Term）	定義
軌跡生成（Rollout）	為評估或學習而產生的任務嘗試或軌跡。
軌跡（Trajectory）	一連串觀察、動作及相關資訊。
後訓練（Post-training）	基礎模型開發後，用以改變行為或能力的訓練。
獎勵（Reward）	學習過程中評估一次嘗試的訊號。
驗證證明（Attestation）	關於執行環境及實測軟體狀態的證據。

術語（Term）	定義
形式驗證（Formal Verification）	在形式系統與假設內，檢查指定性質。
校準（Calibration）	預測機率與觀察頻率的一致程度。
顧客貢獻（Customer Contribution）	營收扣除所採貢獻定義涵蓋的成本。
預測顧客終身價值（Predicted CLV）	在明確或模型化期間內，對未來顧客貢獻的估計。
增量效果（Incrementality）	介入相對於未介入反事實（Counterfactual）所造成的結果差異。
增益（Uplift）	處理效果概念，不只是購買或回應機率。
留出樣本 / 對照組（Holdout）	為明確用途保留、與其他資料分開的比較群或評估集。
資料洩漏（Data Leakage）	使用真實預測時點尚無法取得的資訊。
漂移（Drift）	資料、關係或效能隨時間改變。
付費利用率（Paid Utilization）	明定容量中產生付費使用的比例，不同於技術活動量。

附錄B 採訪方法與證據

本書為李志傑 博士（Dr. Cervantes Lee）的原創現場報告，依2026年11月12至13日舊金山Fort Mason之SGLang Summit周邊照片、場次紀錄與筆記撰寫。Ghirardelli Square的Traction Layer聚會另標為周邊活動。官方議程、公司與技術文件支持場次身分、產品規格及部分公司事實。

本報告可獨立閱讀，章節包含技術解說、相關案例與計算範例。作者錄音、筆記及製作文件保留為私人採訪檔案。編號註釋指出各主題場次證據及原始公開文件，不需閱讀未公開工作報告。

來源歸屬決定陳述能證明什麼。照片記錄可見內容；未獨立測試的講者結果仍是講者報告；產品文件描述參考配置。分析由此推論，不把贊助、募資、示範或路線圖轉成商業績效證據。

技術與公開來源反映2026年11月採訪期材料。獨立完整版重組編輯紀錄，不宣稱重新外部查證或重播每份原錄音。線上文件可能在出版後改變；未確認規格、身分或測量，正文保留限制。

餐飲母體、顧客價值、活動比較、端到端時間及利用率敏感度，均為GII示範計算。未取得或分析真實餐飲顧客資料庫。表格明列假設，讓讀者以實測營運資料替換。

照片保留紀錄內容，裁切、方向調整及編號只改善閱讀，不重建隱藏設備或缺漏投影片文字。官方肖像與原創解說圖均標示；識別證QR憑證已遮蔽。部分投影片仍有遮擋，因原照片即如此。

這些區分決定讀者可賦予證據的權重，適用全書。

表23｜證據區分

項目	本版處理方式
英特爾主管照片	標示現場螢幕轉播，不重建演講引言。
AMD官方肖像	標示議程肖像，不冒充作者拍攝的舞台照片。
ApprenticeBench肖像	使用場次標題圖，避免額外斷言肖像身分。
Traction Layer	標示Ghirardelli Square周邊活動。
Uni³Sim	視為學術AI基礎設施模擬研究。
輝達托盤識別	區分交換器托盤與運算托盤。
Axiom Putnam成果	依Axiom報告：比賽期間八題，最終12題。
PD 75 / 25案例	引擎時間配置，不是詞元比例或通用硬體配方。
顧客母體	假設總數30萬、資料庫10萬檔案，未分析真實資料。
顧客價值案例	明列假設的示範算術，不是公司預測。
基準測試（Benchmarks）	除明標GII教學算例外，均為講者或公司報告。

編輯與出版紀錄

獨立完整版2.0，2026年11月。全球創新研究院李志傑 博士（Dr. Cervantes Lee）採訪、攝影與分析。本版將產業案例整合進論證，以場次歸屬及公開來源註釋取代私人工作文件引用。AI工具協助整理、撰稿及文件製作。更正須指明段落與支持證據；後續發展應另標日期，與11月12至13日現場紀錄區分。

文末註釋與公開參考資料（Endnotes and Public References）

方括號數字對應下列紀錄。會議主張由作者現場照片及筆記支持；部分原始公開文件支持身分、規格與背景。公開技術連結提供脈絡，不獨立驗證講者基準。峰會官方議程為共同活動參考。

SGLang Summit 2026——官方議程與講者名單（Official Agenda and Speaker Roster）
https://www.sglang.io/summit

1. Horace He：關於推論的兩種思考模型（Two Mental Models About Inference）

2026年11月13日SGLang Summit，作者照片與口述紀錄。75%預填充 / 25%解碼是引擎處理時間，不是詞元占比。單人串流四倍快假設資源池平衡及解碼批次相容，示範條件總輸出仍每秒160詞元。第三章公式及算術是教學解讀，不是獨立重現基準。

SGLang預填充與解碼分離（PD Disaggregation）
https://docs.sglang.io/docs/advanced_features/pd_disaggregation

vLLM分離式預填充（Disaggregated Prefilling）
https://docs.vllm.ai/en/latest/features/disagg_prefill/

Modular推論手冊——PD分離（Inference Handbook）
https://handbook.modular.com/inference-optimization/prefill-decode-disaggregation/

Google SRE——服務水準目標（Service-level Objectives）
https://sre.google/sre-book/service-level-objectives/

2. GII研究系列與原創分析

李志傑 博士（Dr. Cervantes Lee），《餐廳作為實體人工智慧平台（The Restaurant as a Physical AI Platform）》，全球創新研究院，2026年10月。第一集提供據點 / 任務 / 網路架構與本書視覺設計。第一章已解釋架構，前集為選讀。餐飲母體、活動計算、營收敏感度及明列示範的案例，均為本書原創分析練習。

3. SGLang、Miles與開場生態系

開場資料列Lianmin Zheng、Ying Sheng與Banghua Zhu，官方肖像於圖說標示。作者場次紀錄及原始專案文件支持模型服務與RL關係，並區分開源專案、商業服務及聯盟治理。

SGLang程式碼儲存庫（Repository）
https://github.com/sgl-project/sglang

Miles程式碼儲存庫（Repository）
https://github.com/radixark/miles

RadixArk——Miles v0.1
https://www.radixark.com/blog/announcing-miles-v0-1

LMSYS——SGLang聯盟（Consortium）
https://www.lmsys.org/blog/2026-11-12-sglang-consortium/

4. 英特爾：主管參與與開放推論承諾

作者於2026年11月12日Fort Mason拍攝的螢幕轉播與觀眾照片，記錄陳立武（Lip-Bu Tan）的可見參與；採訪檔案無完整談話逐字稿。合作陳述來自英特爾公開公告，正文動機及商業意涵為作者解讀。

英特爾與RadixArk合作公告（Collaboration Announcement）
https://www.intc.com/news-events/press-releases/detail/1787/intel-radixark-collaborate-to-advance-open-ai-inference-on

英特爾——代理式AI的開放基礎設施（Open Infrastructure for Agentic AI）
https://www.intc.com/news-events/press-releases/detail/1783/intel-accelerates-open-infrastructure-for-the-agentic-ai-era

LMSYS——SGLang聯盟（Consortium）
https://www.lmsys.org/blog/2026-11-12-sglang-consortium/

5. AMD參與與開發者採用

2026年11月12至13日SGLang Summit，依作者展場照片、官方贊助名單及議程。Vamsi Boppana由官方資料確認，肖像不冒充作者舞台攝影。議程記錄主管、工程及工作坊參與；無完整AMD場次錄音可供重建發言。

SGLang Summit官方議程（Official Agenda）
https://www.sglang.io/summit

AMD人工智慧開發者計畫（AI Developer Program）
https://developer.amd.com/ai-developer-program/

AMD SGLang技術文件（Documentation）
https://rocm.docs.amd.com/projects/ai-ecosystem/en/latest/inference/sglang.html

SAPPHIRE Radeon AI PRO R9700
https://www.sapphiretech.com/en/commercial/radeon-ai-pro-r9700

6. AMD展場硬體與參考規格

作者於2026年11月12日拍攝硬體。MI350X展品是八模組加速器底板；另一MI355X標牌描述液冷平台。開蓋銀色機箱CPU型號及加速器配置未確認。產品文件供參考規格，照片不是運行測試或完整清單。編號只標可見區域，不重建隱藏硬體。

AMD Instinct MI350X平台（Platform）
https://www.amd.com/en/products/accelerators/instinct/mi350/mi350x/platform.html

AMD Instinct MI355X平台（Platform）
https://www.amd.com/en/products/accelerators/instinct/mi350/mi355x/platform.html

AMD ROCm MI350架構（Architecture）
https://rocmdocs.amd.com/en/latest/reference/gpu-arch/mi350.html

AMD SGLang技術文件（Documentation）
https://rocm.docs.amd.com/projects/ai-ecosystem/en/latest/inference/sglang.html

7. 輝達展場硬體與參考配置

作者於2026年11月12至13日Fort Mason峰會拍攝，編號為解說標註。可讀托盤標牌支持GB300、Rubin與Vera識別；整架正面圖不足確定世代。CPU / GPU數是引用參考配置，不是獨立檢查隱藏封裝所得；未取得採購報價或運行基準。

輝達（NVIDIA）GB300 NVL72
https://www.nvidia.com/en-us/data-center/gb300-nvl72/

輝達DGX SuperPOD GB300參考架構（Reference Architecture）
https://docs.nvidia.com/pdf/dgx-spod-gb300-ra.pdf

輝達（NVIDIA）Vera Rubin NVL72
https://www.nvidia.com/en-us/data-center/vera-rubin-nvl72/

輝達DGX Vera Rubin硬體指南（Hardware Guide）
https://docs.nvidia.com/dgx/dgxvr-user-guide/hardware.html

8. Adarsh Shirawalmath，Hugging Face：Beyond the Brrr

2026年11月13日SGLang Summit，Decode Room。作者場次照片、口述紀錄及講者簡報支持核心封裝與回退行為討論。7.5倍比較只適用展示基準；Kernel Hub文件提供封裝與相容性解說。

Hugging Face kernels技術文件（Documentation）
https://huggingface.co/docs/kernels/

Hugging Face kernels儲存庫（Repository）
https://github.com/huggingface/kernels

Kernel builder設計概覽（Design Overview）
https://huggingface.co/docs/kernels/main/en/builder/design-overview

9. IBM Spyre：整合至SGLang

2026年11月13日SGLang Summit。作者照片與紀錄支持整合敘事；不到一週指接入既有硬體支援，不是晶片或整套編譯器開發時間。PyTorch文章及實作儲存庫提供原始技術背景。

PyTorch——以AI在一週內將SGLang整合至Spyre（Integrating SGLang into Spyre in Under a Week with AI）
https://pytorch.org/blog/integrating-sglang-into-spyre-in-under-a-week-with-ai/

PyTorch——將Spyre建為原生PyTorch裝置（Building Spyre as a Native PyTorch Device）
https://pytorch.org/blog/building-spyre-as-a-native-pytorch-device/

torch-spyre / sglang-spyre
https://github.com/torch-spyre/sglang-spyre

10. Felix Wang，Generalist：GEN-1.5與實體提示（Physical Prompting）

2026年11月13日太平洋時間下午1:00至1:15，SGLang Summit，Decode Room。作者照片及口述紀錄，講者為Generalist技術人員。示範與語言模型發展類比為講者主張；區分上下文條件化與參數更新，不從機器人範例推論已部署餐廳。

Generalist——GEN-1.5
https://generalistai.com/blog/gen-1.5

Generalist——超越世界模型（Beyond World Models）
https://generalistai.com/blog/beyond-world-models

Generalist——GEN-1
https://generalistai.com/blog/gen-1

11. Zhichen Zeng，RadixArk：以Miles進行正式環境等級後訓練

2026年11月13日太平洋時間上午11:10至11:25，HiCache Room。作者照片與口述紀錄，主題為軌跡生成、歷史、同步、一致性及恢復。使用某應用執行框架，不證明Miles訓練該框架相關的專有模型。

Miles技術文件（Documentation）
https://miles.radixark.com/docs

Miles儲存庫（Repository）
https://github.com/radixark/miles

Miles研究論文（Paper），arXiv:2609.08368
https://arxiv.org/abs/2609.08368

12. Lucas Atkins，Arcee AI：快速詞元，緩慢代理（Fast Tokens, Slow Agents）

2026年11月13日，Decode Room。作者照片及口述紀錄；Atkins為技術長兼共同創辦人。場次區分生成速度與已驗證任務完成；60秒流程及1.2倍整體加速是本報告原創算例，不是Arcee基準。

Arcee AI公司背景（Company Background）
https://www.arcee.ai/about

Lucas Atkins講者簡介（Speaker Biography）
https://ai.engineer/speakers/lucas-atkins

13. Francis Zhang，Core Automation：研究用推論系統

《為研究時代設計推論系統（Designing Inference Systems for the Age of Research）》，2026年11月13日太平洋時間上午10:45至11:00，HiCache Room。作者照片及口述紀錄。快取介面與記憶體池是工程論述，不證明通用實測加速。

SGLang——統一基數快取（Unified Radix Cache）
https://www.sglang.io/blog/unified-radix-cache

vLLM混合KV快取管理器（Hybrid KV-cache Manager）
https://docs.vllm.ai/en/latest/design/hybrid_kv_cache_manager/

Transformers快取說明（Cache Explanation）
https://huggingface.co/docs/transformers/cache_explanation

14. Tri Dao與Alexander Korte，Baseten：SGL超高速推論（Ultrafast Inference on SGL）

2026年11月13日太平洋時間上午10:30至10:45，HiCache Room。作者照片與口述紀錄。活動及專業檔案識別Dao為Baseten工程師，與同名FlashAttention研究者不同。微秒層結果、解碼步驟毫秒及使用者詞元速率是不同測量；效能主張歸於演講。

Baseten——Flash模型推論優化（Flash-model Inference Optimization）
https://www.baseten.co/blog/how-to-make-flash-models-even-faster-optimizing-deepseek-v41-flash-1017-fast/

Tri Dao專業檔案（Professional Profile）
https://www.linkedin.com/in/tri-dao-03
LinkedIn 可能需要登入；公開備用檔案（GitHub）：https://github.com/dmtri35

峰會官方講者名單（Official Speaker Roster）
https://www.sglang.io/summit/speakers

15. Hang Yin，Phala：機密AI推論與dstack

2026年11月13日SGLang Summit。作者照片及口述紀錄；Yin為共同創辦人兼技術長。驗證證明、可重現建置及受保護運作針對明列信任模型，不驗證AI答案正確，也不消除所有應用層風險。

dstack儲存庫（Repository）
https://github.com/Dstack-TEE/dstack

私密AI閘道（Private AI Gateway）
https://github.com/Dstack-TEE/private-ai-gateway

微軟機密GPU技術文件（Microsoft Confidential GPU Documentation）
https://learn.microsoft.com/en-us/azure/confidential-computing/gpu-options

16. WiCi InferSSD：本地推論與專家權重串流

SGLang Summit 2026，依作者投影片照片及口述紀錄，說明從SSD串流選定MoE權重。每秒7.27詞元限特定配置。容量擴大不代表SSD等同GPU記憶體，也不證明普遍達到雲端同等效能。

WiCi InferSSD技術文章（Technical Article）
https://wici.ai/article/sglang-ssd-expert-pack

InferSSD儲存庫（Repository）
https://github.com/wici-ai/inferssd

WiCi推論產品（Inference Product）
https://wici.ai/wici-infer

17. Traction Layer：Dim Sum and Demo

2026年11月12日舊金山Ghirardelli Square周邊聚會，與峰會舞台議程分開。作者照片及對話紀錄含Shankaran Sitarama與Rahul Kundapur Ramesh。排程結果是有取捨的示範，不是獨立重測基準；商業方案與調校器描述來自Traction Layer資料。

Traction Layer產品（Product）
https://www.tractionlayer.ai/product/

Traction Layer效能（Performance）
https://www.tractionlayer.ai/performance/

SGLang排程器實作（Scheduler Implementation）
https://github.com/sgl-project/sglang/blob/main/python/sglang/srt/managers/scheduler.py

18. Nikita Lazarev，Netpreme：GPU記憶體分層

2026年11月13日太平洋時間下午2:10至2:25，Decode Room。作者投影片照片與口述紀錄；Lazarev為共同創辦人兼技術長。容量及頻寬由講者報告，各保留其範圍，外部容量不視為等同本地HBM。

Netpreme——XM-Sparse、HiSparse與XMem
https://netpreme.com/blog/xm-sparse-sglang-hisparse-netpreme-xmem

Netpreme
https://netpreme.com/

背景參考：DeepSeek-V2 研究論文（Background Paper），arXiv:2405.04434
https://arxiv.org/abs/2405.04434

19. Xibo Sun，騰訊TACO：資源高效推論

SGLang Summit 2026，作者照片與口述支持資源管理、快取介面及相容性敘事；產品與軟體文件提供技術背景。狀態容量擴大、共享正確性及實測服務效能，仍屬不同主張。

騰訊TACO產品（Tencent TACO Product）
https://cloud.tencent.cn/product/taco

FlexKV儲存庫（Repository）
https://github.com/taco-project/FlexKV/blob/main/README.md

SGLang統一快取元件（Unified-cache Components）
https://github.com/sgl-project/sglang/blob/main/python/sglang/srt/mem_cache/unified_cache/components/README.md

20. Zhenhua Zhu：Uni³Sim

2026年11月13日太平洋時間下午3:30至3:45，Decode Room。作者照片及口述識別為清華大學深圳國際研究生院相關專案。模擬結果由講者報告，儲存庫未稽核、預測未獨立重現。主題為AI基礎設施，不是已獨立成立的機器人企業。

學術簡介（Academic Biography）
https://fact-lab.hkust.edu.hk/authors/zhenhua-zhu/

峰會官方議程（Official Summit Agenda）
https://www.sglang.io/summit

21. Palo Alto Networks：即時安全的小型語言模型

Matthew Schulz與Zhemin Su，2026年11月13日太平洋時間下午3:45至4:00，Decode Room。作者投影片照片及口述紀錄。單GPU比較在三個同時客戶端下，p99延遲53.1對44.2毫秒，目標50毫秒；不衡量安全準確率或全機群吞吐量。議題紀錄提供實作背景，不獨立驗證完整基準。

SGLang議題（Issue）43930
https://github.com/sgl-project/sglang/issues/43930

SGLang議題（Issue）43929
https://github.com/sgl-project/sglang/issues/43929

RayService技術文件（Documentation）
https://docs.ray.io/en/latest/cluster/kubernetes/user-guides/rayservice.html

22. Yağız Çalık，ai&：跨異質GPU強化學習

2026年11月13日太平洋時間上午10:15至10:30，HiCache Room。作者照片及口述紀錄。48張GPU、四種裝置及兩家供應商為講者報告；吞吐量及閒置改善保留測試脈絡，不獨立證明模型能力改善。

ai&訓練（Training）
https://www.aiand.com/training

Miles合併請求（Pull Request）3690
https://github.com/radixark/miles/pull/3690

SGLang合併請求（Pull Request）48241
https://github.com/sgl-project/sglang/pull/48241

PyTorch——Miles後訓練堆疊（Post-training Stack）
https://pytorch.org/blog/miles-a-pytorch-native-stack-for-large-scale-llm-rl-post-training/

23. Xiang Deng，NeoCognition：ApprenticeBench

2026年11月13日SGLang Summit。作者照片與口述紀錄。模擬營建會計以歷史、指令及回饋測試持續工作。示範播放時間、內部任務時間與成本計數不同；基準提升不證明自主正式會計可行。

ApprenticeBench
https://apprenticebench.com/

NeoCognition
https://neocognition.io/

Xiang Deng簡介（Biography）
https://xiang-deng.github.io/

24. Tobias Gessler與Rishi Malhotra，Axiom Math

《形式化全部數學的訓練系統（Training Systems for Formalizing All of Mathematics）》，2026年11月13日SGLang Summit。作者投影片照片與口述支持證明搜尋、Lean及形式化敘事；公司專案公告與募資頁提供揭露。Putnam八題在賽期內、四題之後，保留不同時間主張。募資及估值是有日期的揭露，不是營收、現金或商業採用證據。

Axiom——募資揭露（Fundraising Disclosures）
https://axiommath.ai/fundraising/

Axiom——Stacks專案形式化（The Stacks Project Formalized）
https://axiommath.ai/research/the-stacks-project-formalized/

Axiom——龐大定理形式化（The Enormous Theorem Formalized）
https://axiommath.ai/research/the-enormous-theorem-formalized/

Lean語言與證明輔助工具（Language and Proof Assistant）
https://lean-lang.org/

25. Simon Shaolei Du，Apodex：深度發現與預測

2026年11月13日SGLang Summit。作者照片與口述紀錄。多代理推理及足球例歸於演講，不是事後重建的獨立預測。單一案例選少數答案，不構成普遍規則。

Apodex簡介（Introduction）
https://www.apodex.com/blog/apodex

Apodex 1.0
https://www.apodex.com/blog/apodex-1.0

Apodex模型（Models）
https://huggingface.co/apodex

Simon Du學術簡介（Academic Biography）
https://www.cs.washington.edu/people/faculty/simon-du/

26. Sangwu Lee，Krea：擴展擴散模型的強化學習（Scaling RL for Diffusion Models）

官方議程：2026年11月13日太平洋時間下午5:10至5:25，Decode Room。依議程及主題識別，不以人臉辨認。作者照片與口述支持獎勵設計範例；產品及技術報告提供背景，高獎勵分不視為完成圖像可接受的證明。

Krea 3技術報告（Technical Report）
https://www.krea.ai/blog/krea-3-technical-report

Krea 3
https://www.krea.ai/krea-3

峰會官方議程（Official Summit Agenda）
https://www.sglang.io/summit

27. Ryan Lee，MiniMax：多模態系統與代理工作流程

2026年11月SGLang Summit，依議程及主題確認場次。作者投影片照片支持流程敘事，MiniMax原始產品及技術資料提供公司背景。英文產品或國際可用性，本身不證明可量化的美國營收貢獻。

MiniMax公司簡介（Company Profile）
https://ir.minimax.io/corporate-information/company-profile

MiniMax M3.1F
https://www.minimax.io/blog/minimax-m31f

MiniMax H3
https://www.minimax.io/blog/minimax-h3

SGLang——MiniMax M3.1 Flash支援
https://www.sglang.io/blog/minimax-m3.1-flash-day0-support

28. Ethan Anderson，SF Compute：AI利潤可能留在哪裡

2026年11月13日太平洋時間下午2:40至2:55，Decode Room。作者照片與口述。矩陣檢視運算需求及模型供給組合，是條件架構，不是無條件預測；未重建被遮住投影片文字。

SF Compute
https://sfcompute.com/

SF Compute技術文件（Documentation）
https://docs.sfcompute.com/

SF Compute公司背景（Company Background）
https://sfcompute.com/about

29. Jasper Zhang，Hyperbolic：全球AI運算網格

2026年11月13日SGLang Summit。作者照片與口述；Zhang為共同創辦人兼執行長。1,000張GPU預訂與轉售回收算術解釋示範情境，不是報價或已核實顧客節省。服務可用性及推出進度歸於講者。

Hyperbolic——公司方向（Company Direction）
https://www.hyperbolic.ai/blog/hyperbolic-next-chapter

Hyperbolic——融資公告（Financing Announcement）
https://www.hyperbolic.ai/blog/hyperbolic-secures-20m-total-funding-with-latest-series-a

30. Zhanghao Wu，SkyPilot：強化學習基礎設施

2026年11月13日SGLang Summit。作者照片與口述，討論軌跡生成、訓練及支援服務協調。利用率與測試結果限展示配置；公司技術文章及文件補充流程細節。

SkyPilot——強化學習無所不在（RL Is Everything Everywhere All at Once）
https://skypilot.ai/blog/rl-everything

SkyPilot——強化學習受推論瓶頸限制（RL Is Bottlenecked by Inference）
https://skypilot.ai/blog/slime-agentic-rl

SkyPilot——Grace Blackwell需要新排程器（Grace Blackwell Needs a New Scheduler）
https://skypilot.ai/blog/grace-blackwell-needs-a-new-scheduler

工作群組（Job Groups）技術文件
https://docs.skypilot.ai/en/master/examples/job-groups.html