**Bảng So Sánh Chi Tiết Các Công Cụ/Mô Hình AI**

**Lưu ý quan trọng:**

* Mô hình (**Models**) vs. Công cụ (**Tools**/**Platforms**): Có sự khác biệt lớn giữa một "mô hình" (như **Vicuna**, **Llama**-**2**) và một "công cụ" hoặc "nền tảng" (như Ollama, LM Studio).

Các *mô hình models AI bản thân chúng không có giao diện người dùng* hay khả năng tải/chuyển đổi, chúng cần một công cụ để chạy. Bảng này sẽ làm rõ điều đó.

* **GGLF**: Tôi giả định bạn muốn nói đến định dạng **GGUF** (**GPT-Generated Unified Format**), một định dạng phổ biến cho các LLM chạy cục bộ, đặc biệt là với llama.cpp.
* "**llm**" và "**qwen" trong cột 3:** "llm" quá chung chung. Tôi sẽ hiểu là khả năng tương thích với các định dạng LLM phổ biến khác. "qwen" là tên của một dòng mô hình, tôi sẽ hiểu là khả năng xử lý định dạng native của Qwen hoặc các dạng đã được chuyển đổi. "dataset" sẽ đề cập đến khả năng xử lý dữ liệu cho fine-tuning/RAG.

*Dưới đây là bảng so sánh chi tiết các công cụ và mô hình bạn đã liệt kê, dựa trên 5 cột thông tin yêu cầu.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tiêu Chí | Ollama agent | Jan.ia | LM Studio | marimo notebook | jupyter notebook |
| 1. Tính năng, Chức năng, Khả năng tùy biến prompt, mở rộng điều chỉnh tham số chuyển thành Form fields | **Chức năng chính:** Serve LLMs cục bộ với API. **Tùy biến:** Tùy biến prompt thông qua Modelfile (system prompt, template). Tham số qua API. UI (như Open WebUI) thêm form. | **Chức năng chính:** Ứng dụng desktop chạy LLM. **Tùy biến:** UI thân thiện, chatbox, prompt templates, điều chỉnh tham số cơ bản (nhiệt độ, top\_k). | **Chức năng chính:** Ứng dụng desktop tải, chạy LLM. **Tùy biến:** UI trực quan, chatbox, nhiều tùy chọn prompt, điều chỉnh tham số chi tiết qua sliders/fields. | **Chức năng chính:** Notebook reactive, xây dựng app web. **Tùy biến:** Hoàn toàn tùy biến bằng Python, có thể tạo form input động cho prompt/params. | **Chức năng chính:** Môi trường code tương tác. **Tùy biến:** Hoàn toàn tùy biến bằng Python, có thể dùng ipywidgets tạo form cho prompt/params. |
| 2. Tải models từ hãng: huggingface, kaggle, openai, ollama, anthorpic, qwen, deepseek | **Ollama Hub:** Tải trực tiếp các model đã chuyển đổi sang định dạng Ollama (nhiều model từ HF). **Không trực tiếp:** từ HF, Kaggle, OpenAI, Anthropic (API). | **Hugging Face:** Tải GGUF trực tiếp. **Local Files:** Chạy model từ file cục bộ. **Không trực tiếp:** từ Ollama Hub, OpenAI, Anthropic. | **Hugging Face:** Tải GGUF trực tiếp. **Local Files:** Chạy model từ file cục bộ. **Không trực tiếp:** từ Ollama Hub, OpenAI, Anthropic. | **Qua Code:** Có thể tải model từ HF (thư viện transformers), Kaggle (API), OpenAI/Anthropic/Ollama (qua API). | **Qua Code:** Tải model từ HF (thư viện transformers), Kaggle (API), OpenAI/Anthropic/Ollama (qua API). |
| 3. GGLF, llm, qwen, dataset | **GGUF:** *Sử dụng* GGUF (đã được đóng gói trong định dạng Ollama). Không phải công cụ chuyển đổi. | **GGUF:** *Sử dụng* GGUF. Không phải công cụ chuyển đổi. | **GGUF:** *Sử dụng* GGUF. Không phải công cụ chuyển đổi. | **Dataset:** Có thể xử lý, phân tích dataset. **Transform:** Có thể viết code Python để chuyển đổi định dạng model (ví dụ: PyTorch -> GGUF). | **Dataset:** Có thể xử lý, phân tích dataset. **Transform:** Có thể viết code Python để chuyển đổi định dạng model (ví dụ: PyTorch -> GGUF). |
| 4. Web rest api local, Tích hợp webui các loại AI Agent chạy localhost (Offline) | **Web REST API Local:** CÓ (mặc định có API). **Tích hợp WebUI:** CÓ (thường dùng với Open WebUI). **Offline:** CÓ. | **Web REST API Local:** KHÔNG (là ứng dụng desktop). **Tích hợp WebUI:** KHÔNG. **Offline:** CÓ. | **Web REST API Local:** CÓ (có chế độ OpenAI-compatible API server). **Tích hợp WebUI:** CÓ (UI nội bộ hoặc với các tool hỗ trợ OpenAI API). **Offline:** CÓ. | **Web REST API Local:** CÓ (có thể viết code để expose API). **Tích hợp WebUI:** CÓ (có thể tự xây dựng UI). **Offline:** CÓ. | **Web REST API Local:** CÓ (có thể viết code để expose API). **Tích hợp WebUI:** CÓ (có thể tự xây dựng UI). **Offline:** CÓ. |
| 5. Chạy local trên win 11 pro, win 2019 dc server, linux ubuntu 24.04 | **Windows 11 Pro:** CÓ. **Windows 2019 DC Server:** CÓ. **Linux Ubuntu 24.04:** CÓ. | **Windows 11 Pro:** CÓ. **Windows 2019 DC Server:** CÓ (dưới dạng ứng dụng desktop). **Linux Ubuntu 24.04:** CÓ. | **Windows 11 Pro:** CÓ. **Windows 2019 DC Server:** CÓ (dưới dạng ứng dụng desktop). **Linux Ubuntu 24.04:** CÓ. | **Windows 11 Pro:** CÓ. **Windows 2019 DC Server:** CÓ. **Linux Ubuntu 24.04:** CÓ (yêu cầu Python). | **Windows 11 Pro:** CÓ. **Windows 2019 DC Server:** CÓ. **Linux Ubuntu 24.04:** CÓ (yêu cầu Python). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Open WebUI | Vicuna | Llama-2-7B-Chat-HF | Text Generation WebUI Oobabooga | FastChat |
| Chức năng chính: Web UI cho Ollama/LocalAI. Tùy biến: UI chat hiện đại, prompt templates, system prompt, điều chỉnh tham số qua form/slider. | **Bản thân là model:** Khả năng do model định nghĩa. Tương tác qua công cụ khác (Oobabooga, FastChat). | **Bản thân là model:** Khả năng do model định nghĩa. Tương tác qua công cụ khác (Ollama, LM Studio). | **Chức năng chính:** Web UI toàn diện cho text generation. **Tùy biến:** Rất mạnh mẽ, nhiều chế độ chat, prompt templates, tùy chỉnh hàng loạt tham số (nhiệt độ, top\_p, top\_k, repetition penalty, Mirostat, v.v.), hỗ trợ extensions. | **Chức năng chính:** Framework huấn luyện/phục vụ LLM. **Tùy biến:** Web UI chat đơn giản, prompt cơ bản, vài tham số thông qua UI/API. |
| Không trực tiếp: Là UI. Tải model qua các service nó kết nối (Ollama, LocalAI, OpenAI API). | **Hugging Face:** Là nơi chứa model này (thường là PyTorch, Safetensors). Cần công cụ để tải/chạy. | **Hugging Face:** Là nơi chứa model này (thường là PyTorch, Safetensors). Cần công cụ để tải/chạy. | **Hugging Face:** Tải trực tiếp nhiều định dạng (GGUF, Safetensors, PyTorch). **Local Files:** Chạy từ file cục bộ. | **Hugging Face:** Tải các model tương thích. **Không trực tiếp:** Ollama, OpenAI, Anthropic. |
| Transform: Không, nó là UI. | **Bản thân là model:** Tồn tại ở định dạng PyTorch/Safetensors trên HF. Cần công cụ khác để chuyển sang GGUF. | **Bản thân là model:** Tồn tại ở định dạng PyTorch/Safetensors trên HF. Cần công cụ khác để chuyển sang GGUF. | **GGUF/khác:** *Hỗ trợ tải và chạy* nhiều định dạng (GGUF, PyTorch, Safetensors). Có thể có script chuyển đổi đi kèm. | **LLM formats:** Hỗ trợ các định dạng model của Hugging Face (PyTorch, Safetensors). Không chuyển đổi. |
| Web REST API Local: KHÔNG (chỉ là UI). Tích hợp WebUI: CÓ (chính nó là WebUI). Offline: CÓ (khi kết nối với Ollama/LocalAI offline). | **Bản thân là model:** KHÔNG. Cần một công cụ để phục vụ API/UI. | **Bản thân là model:** KHÔNG. Cần một công cụ để phục vụ API/UI. | **Web REST API Local:** CÓ (có chế độ OpenAI-compatible API server). **Tích hợp WebUI:** CÓ (chính nó là WebUI). **Offline:** CÓ. | **Web REST API Local:** CÓ (có API server). **Tích hợp WebUI:** CÓ (có WebUI mẫu). **Offline:** CÓ. |
| Windows 11 Pro: CÓ (Docker/Python). Windows 2019 DC Server: CÓ (Docker/Python). Linux Ubuntu 24.04: CÓ (Docker/Python). | **Bản thân là model:** CÓ (chạy trên bất kỳ OS nào thông qua một framework phục vụ). | **Bản thân là model:** CÓ (chạy trên bất kỳ OS nào thông qua một framework phục vụ). | **Windows 11 Pro:** CÓ (Python). **Windows 2019 DC Server:** CÓ (Python). **Linux Ubuntu 24.04:** CÓ (Python). | **Windows 11 Pro:** CÓ (Python). **Windows 2019 DC Server:** CÓ (Python). **Linux Ubuntu 24.04:** CÓ (Python). |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| litellm | anythingllm | llama.cpp | PrivateGPT | LocalAI |
| Chức năng chính: Proxy API thống nhất cho nhiều LLM. Tùy biến: Ánh xạ tham số giữa các API, cho phép tùy biến prompt theo chuẩn API gốc. Không có UI. | **Chức năng chính:** Nền tảng RAG, Chatbot. **Tùy biến:** Web UI cho chat, prompt templates, quản lý tài liệu, có thể chỉnh system prompt. | **Chức năng chính:** Core engine inference GGUF. **Tùy biến:** CLI, prompt trực tiếp qua dòng lệnh, tham số cơ bản qua flag. Server mode có API. | **Chức năng chính:** RAG riêng tư. **Tùy biến:** Web UI cho chat, tải tài liệu. Prompt thông qua API/UI, chỉnh system prompt. | **Chức năng chính:** Serve LLMs (OpenAI API-compatible). **Tùy biến:** API-centric, tùy biến prompt và tham số thông qua request body JSON. |
| Proxy: Không tải, mà làm proxy cho API của OpenAI, Anthropic, Cohere, Azure, Ollama, Hugging Face Inference API. | **Tích hợp API/Local:** OpenAI, Anthropic, Hugging Face (API), Ollama, LocalAI. **Không tải trực tiếp:** từ Kaggle. | **Không tải:** Nó là engine chạy GGUF. Cần tải file GGUF từ nguồn khác (ví dụ: HF). | **Tích hợp API/Local:** OpenAI, Anthropic (API), Hugging Face (local/API), Ollama, LocalAI. | **Tải/Chạy:** Hỗ trợ tải và chạy các model từ Hugging Face (GGUF, Safetensors, Pytorch), Ollama. |
| Transform: Không, nó xử lý API requests/responses. Không liên quan đến định dạng model file hay dataset. | **Dataset:** Xử lý dataset cho RAG (tải tài liệu, tạo embeddings). Không transform model. | **GGUF:** *Là* tiêu chuẩn để chạy GGUF. Có script (convert.py) để chuyển đổi model PyTorch sang GGUF. | **Dataset:** Xử lý dataset cho RAG (tải tài liệu, tạo embeddings). Không transform model. | **GGUF/khác:** *Hỗ trợ chạy* nhiều định dạng (GGUF, PyTorch, ONNX, TF). Không phải công cụ chuyển đổi chính. |
| Web REST API Local: CÓ (chính nó là một proxy API server). Tích hợp WebUI: CÓ (nếu webUI kết nối tới API của nó). Offline: CÓ (nếu backend LLM cũng offline). | **Web REST API Local:** CÓ (có API internal và external). **Tích hợp WebUI:** CÓ (có sẵn WebUI mạnh mẽ). **Offline:** CÓ. | **Web REST API Local:** CÓ (với llama.cpp/server). **Tích hợp WebUI:** CÓ (với các UI hỗ trợ OpenAI API hoặc llama.cpp API). **Offline:** CÓ. | **Web REST API Local:** CÓ (có API internal). **Tích hợp WebUI:** CÓ (có sẵn WebUI). **Offline:** CÓ. | **Web REST API Local:** CÓ (chính nó là một OpenAI-compatible API server). **Tích hợp WebUI:** CÓ (với Open WebUI, anythingLLM, v.v.). **Offline:** CÓ. |
| Windows 11 Pro: CÓ. Windows 2019 DC Server: CÓ. Linux Ubuntu 24.04: CÓ (yêu cầu Python). | **Windows 11 Pro:** CÓ (Docker/Native). **Windows 2019 DC Server:** CÓ (Docker/Native). **Linux Ubuntu 24.04:** CÓ (Docker/Native). | **Windows 11 Pro:** CÓ. **Windows 2019 DC Server:** CÓ. **Linux Ubuntu 24.04:** CÓ (biên dịch). | **Windows 11 Pro:** CÓ (Docker/Python). **Windows 2019 DC Server:** CÓ (Docker/Python). **Linux Ubuntu 24.04:** CÓ (Docker/Python). | **Windows 11 Pro:** CÓ (Docker/Native). **Windows 2019 DC Server:** CÓ (Docker/Native). **Linux Ubuntu 24.04:** CÓ (Docker/Native). |

**Bảng So Sánh Các Công Cụ LLM và Môi Trường Lập Trình**

Để lập bảng so sánh này một cách chính xác, cần lưu ý rằng tốc độ và chất lượng trả lời thường phụ thuộc rất nhiều vào:

1. **Phần cứng (Hardware)**: CPU, GPU, RAM của máy tính bạn đang chạy.

2. **Mô hình LLM (LLM Model)**: Kích thước, kiến trúc, và chất lượng huấn luyện của mô hình bạn sử dụng (ví dụ: Llama 3 8B, 70B, Mixtral, Code Llama, v.v.).

3. **Prompt:** Độ dài và độ phức tạp của câu hỏi/yêu cầu.

4. **Cấu hình:** Các thiết lập tối ưu hóa của từng phần mềm.

Vì vậy, các con số "ms" sẽ mang tính chất ước lượng rất rộng hoặc là mô tả định tính (nhanh, trung bình, chậm) trong điều kiện tối ưu (phần cứng mạnh, mô hình phù hợp). "MS" cho tốc độ chat/phản hồi thường đo từ lúc gửi prompt đến lúc nhận token đầu tiên hoặc hoàn thành.

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí / Công cụ | 1. Tốc độ chat (ước lượng) | 2. Tốc độ code chuyển từ text/keyword search sang SQL/DAX & Debug | 3. Chất lượng trả lời |
| Ollama (và Ollama Agent) | **Hồi đáp:** Rất nhanh (vài trăm ms - vài giây cho prompt ngắn trên GPU mạnh). **Upload data nguồn:** Rất nhanh (tính theo tốc độ đọc context vào LLM, vài ms - vài trăm ms). **Soạn code Python (Long context):** Nhanh (vài giây - vài chục giây tùy độ dài và mô hình). | **Tốc độ chuyển đổi:** Nhanh (vài giây - vài chục giây tùy độ phức tạp của yêu cầu và mô hình). **SQL/DAX:** Khả năng phụ thuộc hoàn toàn vào mô hình. Các mô hình base thường cần fine-tune hoặc prompt engineering tốt. **Debug/Gợi ý:** Hạn chế trực tiếp, thường chỉ gợi ý text. Kiểm tra cần thực hiện thủ công trong công cụ riêng. | **Phụ thuộc mô hình:** Hoàn toàn dựa vào chất lượng của mô hình bạn chọn (Llama 3, Mixtral, Code Llama, v.v.). **Ngữ cảnh/Sự thật:** Tốt nếu mô hình được huấn luyện tốt và prompt rõ ràng. Có thể "hallucinate". **Tham chiếu:** Không có tham chiếu tự động trừ khi mô hình được tích hợp RAG hoặc huấn luyện đặc biệt. |
| Jan.ia | **Hồi đáp:** Nhanh (tương tự Ollama, vài trăm ms - vài giây). **Upload data nguồn:** Rất nhanh (tính theo tốc độ đọc context vào LLM, vài ms - vài trăm ms). **Soạn code Python (Long context):** Nhanh (vài giây - vài chục giây). | **Tốc độ chuyển đổi:** Nhanh (vài giây - vài chục giây tùy độ phức tạp của yêu cầu và mô hình). **SQL/DAX:** Khả năng phụ thuộc hoàn toàn vào mô hình. **Debug/Gợi ý:** Hạn chế, giao diện chat thông thường. Kiểm tra cần thủ công. | **Phụ thuộc mô hình:** Hoàn toàn dựa vào mô hình cục bộ được tải. **Ngữ cảnh/Sự thật:** Tốt nếu mô hình được huấn luyện tốt. **Tham chiếu:** Không có tham chiếu tự động. |
| LM Studio | **Hồi đáp:** Rất nhanh (tương tự Ollama, vài trăm ms - vài giây). **Upload data nguồn:** Rất nhanh (tính theo tốc độ đọc context vào LLM, vài ms - vài trăm ms). **Soạn code Python (Long context):** Nhanh (vài giây - vài chục giây). | **Tốc độ chuyển đổi:** Nhanh (vài giây - vài chục giây tùy độ phức tạp của yêu cầu và mô hình). **SQL/DAX:** Khả năng phụ thuộc hoàn toàn vào mô hình. **Debug/Gợi ý:** Hạn chế, giao diện chat thông thường. Kiểm tra cần thủ công. | **Phụ thuộc mô hình:** Hoàn toàn dựa vào mô hình cục bộ được tải. **Ngữ cảnh/Sự thật:** Tốt nếu mô hình được huấn luyện tốt. **Tham chiếu:** Không có tham chiếu tự động. |
| Marimo Notebook | **Hồi đáp:** **Không phải công cụ chat LLM trực tiếp.** Tốc độ hiển thị phản hồi từ các đoạn code Python/LLM được tích hợp là nhanh (vài ms cho UI, vài giây nếu gọi LLM API). **Upload data nguồn:** Nhanh (tính theo tốc độ đọc dữ liệu vào Python, vài ms - vài giây cho file nhỏ, vài phút cho file lớn). **Soạn code Python (Long context):** **Không tự động sinh code LLM.** Tốc độ thực thi code Python là rất nhanh. | **Tốc độ chuyển đổi:** **Không tự động sinh code LLM.** Để sinh code SQL/DAX từ text, cần tích hợp LLM (qua API) vào notebook. **SQL/DAX:** Có thể chạy và kiểm tra code SQL/DAX dễ dàng trong môi trường Python sau khi sinh. **Debug/Gợi ý:** Tối ưu cho debug code Python/Data Analysis. Gợi ý từ LLM cần tích hợp riêng. | **Chất lượng:** Phụ thuộc vào chất lượng code do con người viết hoặc LLM được tích hợp. **Ngữ cảnh/Sự thật:** Do người dùng/LLM tích hợp đảm bảo. **Tham chiếu:** Có thể thêm thủ công qua code. |
| Jupyter Notebook | **Hồi đáp:** **Không phải công cụ chat LLM trực tiếp.** Tốc độ hiển thị phản hồi từ các đoạn code Python/LLM được tích hợp là nhanh (vài ms cho UI, vài giây nếu gọi LLM API). **Upload data nguồn:** Nhanh (tính theo tốc độ đọc dữ liệu vào Python, vài ms - vài giây cho file nhỏ, vài phút cho file lớn). **Soạn code Python (Long context):** **Không tự động sinh code LLM.** Tốc độ thực thi code Python là rất nhanh. | **Tốc độ chuyển đổi:** **Không tự động sinh code LLM.** Để sinh code SQL/DAX từ text, cần tích hợp LLM (qua API) vào notebook. **SQL/DAX:** Có thể chạy và kiểm tra code SQL/DAX dễ dàng trong môi trường Python sau khi sinh. **Debug/Gợi ý:** Tối ưu cho debug code Python/Data Analysis. Gợi ý từ LLM cần tích hợp riêng. | **Chất lượng:** Phụ thuộc vào chất lượng code do con người viết hoặc LLM được tích hợp. **Ngữ cảnh/Sự thật:** Do người dùng/LLM tích hợp đảm bảo. **Tham chiếu:** Có thể thêm thủ công qua code. |
| AnythingLLM | **Hồi đáp:** Trung bình - Nhanh (vài giây - vài chục giây). Tốc độ phụ thuộc vào LLM backend (API hay local) và hiệu suất RAG. **Upload data nguồn:** Nhanh (tính theo tốc độ index tài liệu vào Vector DB, vài giây - vài phút tùy dung lượng). **Soạn code Python (Long context):** Khả năng tùy thuộc vào mô hình backend và việc RAG có hỗ trợ context về code hay không. Không phải trọng tâm chính. | **Tốc độ chuyển đổi:** Tương tự tốc độ chat, phụ thuộc LLM backend và RAG. **SQL/DAX:** Khả năng phụ thuộc vào LLM backend và chất lượng dữ liệu được RAG (ví dụ: lược đồ DB, ví dụ query). **Debug/Gợi ý:** Hạn chế. Thường chỉ cung cấp text. | **Chất lượng:** Rất tốt cho các câu hỏi dựa trên dữ liệu nguồn được cung cấp (RAG). **Ngữ cảnh/Sự thật:** Cao do có RAG (truy xuất từ tài liệu nguồn). **Tham chiếu:** Cung cấp tham chiếu (nguồn tài liệu) rất tốt. |
| LocalAI | **Hồi đáp:** Rất nhanh (tương tự Ollama, vài trăm ms - vài giây). **Upload data nguồn:** Rất nhanh (tính theo tốc độ đọc context vào LLM, vài ms - vài trăm ms). **Soạn code Python (Long context):** Nhanh (vài giây - vài chục giây). | **Tốc độ chuyển đổi:** Nhanh (vài giây - vài chục giây tùy độ phức tạp của yêu cầu và mô hình). **SQL/DAX:** Khả năng phụ thuộc hoàn toàn vào mô hình. **Debug/Gợi ý:** Hạn chế. Giao diện API, không có UI trực tiếp. | **Phụ thuộc mô hình:** Hoàn toàn dựa vào mô hình cục bộ được tải. **Ngữ cảnh/Sự thật:** Tốt nếu mô hình được huấn luyện tốt. **Tham chiếu:** Không có tham chiếu tự động. |
| Open WebUI | **Hồi đáp:** Rất nhanh (phụ thuộc vào backend Ollama/LocalAI/API, vài trăm ms - vài giây). **Upload data nguồn:** Rất nhanh (tính theo tốc độ đọc context vào LLM, vài ms - vài trăm ms). Có tích hợp RAG, tốc độ upload dữ liệu nguồn cho RAG tương tự AnythingLLM. **Soạn code Python (Long context):** Nhanh (vài giây - vài chục giây), phụ thuộc backend LLM. | **Tốc độ chuyển đổi:** Nhanh (vài giây - vài chục giây), phụ thuộc backend LLM. **SQL/DAX:** Khả năng phụ thuộc vào mô hình backend. Có thể tích hợp RAG với lược đồ DB để tăng độ chính xác. **Debug/Gợi ý:** Hạn chế, chỉ cung cấp text. Cần kiểm tra thủ công. | **Phụ thuộc mô hình/Backend:** Tốt nếu mô hình backend tốt. **Ngữ cảnh/Sự thật:** Cao nếu có RAG với dữ liệu nguồn. **Tham chiếu:** Có thể hiển thị nguồn nếu RAG được cấu hình. |
| Text Generation WebUI (Oobabooga) | **Hồi đáp:** Rất nhanh (tương tự Ollama, vài trăm ms - vài giây). **Upload data nguồn:** Rất nhanh (tính theo tốc độ đọc context vào LLM, vài ms - vài trăm ms). Có các extension để xử lý dữ liệu phức tạp hơn. **Soạn code Python (Long context):** Nhanh (vài giây - vài chục giây), phụ thuộc mô hình. | **Tốc độ chuyển đổi:** Nhanh (vài giây - vài chục giây), phụ thuộc mô hình. **SQL/DAX:** Khả năng phụ thuộc hoàn toàn vào mô hình. Có thể sử dụng các extension để hỗ trợ specific task. **Debug/Gợi ý:** Hạn chế, giao diện chat thông thường. Kiểm tra cần thủ công. | **Phụ thuộc mô hình:** Rất linh hoạt trong việc tải và thử nghiệm nhiều mô hình khác nhau, nên chất lượng hoàn toàn tùy thuộc vào mô hình được chọn. **Ngữ cảnh/Sự thật:** Tốt nếu mô hình được huấn luyện tốt. **Tham chiếu:** Không có tham chiếu tự động trừ khi mô hình hoặc extension hỗ trợ. |

**Giải thích thêm:**

* **Tốc độ "ms"**: Rất khó để định lượng chính xác "ms" vì nó biến động lớn. Các giá trị trên mang tính tương đối và giả định bạn đang sử dụng phần cứng tốt (có GPU) cho các công cụ chạy LLM cục bộ.
* **Upload data nguồn để xử lý thuật toán dữ liệu ms**: Đối với các công cụ LLM, "upload data" ở đây thường có nghĩa là đưa dữ liệu vào cửa sổ ngữ cảnh (context window) của mô hình. Tốc độ này rất nhanh, tính bằng mili giây hoặc vài giây cho lượng lớn token, nhưng bị giới hạn bởi kích thước context của mô hình. Với Marimo/Jupyter, nó là tốc độ Python đọc dữ liệu vào bộ nhớ.
* **Soạn coder python từ ngữ cảnh dài "Long context" ms**: Tốc độ sinh code Python dài phụ thuộc vào khả năng xử lý long context của mô hình LLM và hiệu suất của phần cứng.
* **SQL/DAX & Debug**: Đây là một tác vụ rất chuyên biệt. Hầu hết các công cụ LLM cơ bản chỉ là giao diện hoặc runtime. Khả năng sinh code SQL/DAX chính xác, tối ưu, và dễ kiểm tra (Power BI, Power Query, DAX Visual) phụ thuộc hoàn toàn vào:
  + **Mô hình LLM:** Cần một mô hình được fine-tune đặc biệt cho code generation, SQL, DAX, hoặc một mô hình base rất mạnh (ví dụ: Llama 3 70B, GPT-4).
  + **Context:** Bạn cung cấp đủ lược đồ cơ sở dữ liệu (schema), ví dụ dữ liệu, hoặc ví dụ query mong muốn.
  + **RAG:** Hệ thống RAG có thể truy xuất các tài liệu hướng dẫn, best practices về SQL/DAX để cung cấp cho LLM.
  + **Công cụ debug:** Hiện tại, hầu hết các công cụ LLM chưa có khả năng debug "nhanh và chính xác" ngay trong giao diện của chúng cho SQL/DAX như một IDE chuyên dụng. Người dùng thường phải copy code sang Power BI/SQL Server Management Studio để kiểm tra.
* **Chất lượng trả lời**: Yếu tố quan trọng nhất. Một giao diện tốt không làm cho mô hình kém chất lượng trở nên thông minh hơn. RAG (Retrieval Augmented Generation) là kỹ thuật quan trọng để tăng độ chính xác và khả năng cung cấp tham chiếu.