**CASE STUDY : SELF-DRIVING CAR**

**MODULE REVIEW 1**

**Tổng quan các cấp độ tự động hóa (SAE Levels) :**

**Level 1**:Hỗ trợ tài xế (ví dụ: hỗ trợ giữ làn, hỗ trợ tốc độ) — tài xế vẫn phải giữ tay lái, cảnh giác liên tục.

**Level 2**:Tự động hóa một phần — xe có thể kiểm soát tay lái và tốc độ trong một số điều kiện, nhưng tài xế phải sẵn sàng can thiệp.

**Level 3**:Tự động có điều kiện — xe có thể xử lý hầu hết tình huống; tài xế có thể lơ đi nhưng phải sẵn sàng chuyển về kiểm soát nếu hệ thống yêu cầu.

**Level 4**:Tự động cao — xe có thể vận hành hoàn toàn trong phạm vi môi trường nhất định mà không cần sự can thiệp của con người.

**Level 5**:Tự động hoàn toàn — bất kể môi trường, xe có thể hoạt động mà không cần con người làm bất kỳ điều gì (“steering wheel optional”).

**Why is this so hard?**

Phát triển xe tự lái không chỉ là ghép các khối “sensing – mapping – planning – control” lại với nhau. Khó khăn đến từ bản chất phức tạp, bất định, và yêu cầu an toàn tuyệt đối của môi trường giao thông thực.

Cảm biến (camera, LiDAR, radar) đều có **hạn chế vật lý**: ánh sáng chói, sương mù, mưa tuyết, vật thể bị che khuất (occlusion). Ngay cả khi kết hợp nhiều cảm biến (sensor fusion), hệ thống vẫn có thể đưa ra nhận diện sai. Một vật thể nhỏ (túi nilon, động vật) có thể bị nhận nhầm gây quyết định nguy hiểm. Tất cả điều trên cho thấy perception từ nhiều nguồn cảm biến vẫn dễ bị sai trong điều kiện thời tiết xấu hoặc trường hợp hiếm.

Bản đồ HD rất chi tiết, nhưng cập nhật liên tục là điều gần như bất khả thi vì đường phố luôn thay đổi (công trình, vạch mới, biển báo mới).Trong khi đó, GPS bị che khuất ở đô thị đông đúc hoặc đường hầm, khiến xe khó xác định chính xác vị trí nếu không có thuật toán SLAM và cảm biến phụ trợ.

Lập kế hoạch đường đi (Path Planning) trong môi trường động là NP-hard. Xe phải ra quyết định trong mili-giây, cân bằng giữa: an toàn, tốc độ, luật giao thông, thoải mái hành khách. Phải dự đoán hành vi người khác (người đi bộ, xe máy, ô tô khác) – điều này cực kỳ khó vì hành vi con người không luôn tuân thủ luật.

Điều khiển (Control) phải tuyệt đối chính xác và có dự phòng. Xe thật chịu tác động từ trễ cơ học, ma sát, mặt đường trơn, lốp mòn dễ làm kế hoạch lý tưởng có thể không thực thi chính xác. Do đó, hệ thống điều khiển cần siêu tin cậy và có redundancy (dự phòng). Nếu phanh hoặc vô-lăng lỗi, hệ thống phải có cơ chế fallback ngay lập tức.

Các yếu tố ngoài kỹ thuật: edge cases, luật pháp, xã hội. Edge cases: trẻ em chạy qua đường, tín hiệu giao thông hỏng, biển báo bị vẽ bậy, xe khác lái ẩu. Đây là những tình huống hiếm nhưng cực kỳ nguy hiểm, khó đưa vào huấn luyện. Luật pháp & trách nhiệm pháp lý: Ai chịu trách nhiệm nếu xe gây tai nạn? Nhà sản xuất, chủ xe hay công ty phần mềm? Chấp nhận xã hội: Một tai nạn nhỏ cũng có thể phá hủy niềm tin của công chúng vào công nghệ.

**If we have two cars and one provides more (expected) utility.Which car is rational?**

Chiếc xe nào ra quyết định nhất quán theo nguyên tắc tối đa hóa expected utility sẽ được coi là rational.

Ví dụ: nếu Car A chọn hành động an toàn hơn, giảm rủi ro tai nạn → utility kỳ vọng cao hơn → Car A rational hơn Car B.

**Can a rational self-driving car be involved in an accident?**

Có. Rationality không đảm bảo “hoàn hảo” mà chỉ đảm bảo hành động tối ưu dựa trên thông tin và mô hình hiện có.

Một xe rational vẫn có thể gặp tai nạn do: thông tin cảm biến thiếu/nhầm lẫn (perception error),hành vi bất định từ xe khác/người đi bộ,“edge cases” chưa từng gặp (được gọi là open-world problem).Nói cách khác, rationality ≠ omniscience.

**How would a self-driving car explore and learn?**

-Sử dụng percepts (camera, lidar, radar, GPS, bản đồ, dữ liệu lái trước đó) để: xây dựng mô hình thế giới, cập nhật kiến thức giao thông, học từ huấn luyện, mô phỏng, trải nghiệm

**What does bounded rationality mean for a self-driving car?**

Bounded rationality (Herbert Simon, Nobel 1978): tác nhân không thể tối đa hóa utility một cách tuyệt đối vì giới hạn về tính toán, thông tin, và thời gian.

Với xe , nghĩa là:

- Xe không thể xét mọi hành động khả dĩ (không đủ thời gian tính toán real-time).

- Xe chỉ có dữ liệu cảm biến giới hạn, bản đồ có thể lỗi thời.

- Do đó, xe dùng heuristics, approximation algorithms, rule-based safety constraints để ra quyết định “tốt đủ” (satisficing), thay vì tối ưu toàn cục

**PEAS Description of the Environment of a Self-Driving Car**

**Performance measure :**

* An toàn
* Tuân thủ luật giao thông
* Thời gian đến đích( tối uu hành trình)
* Tiết kiệm nhiên liệu/ năng lượng
* Sự thoải mái của hành khách
* Tránh kẹt xe, giảm chi phí bảo trì.

**Environment:**

* Đường xá (cao tốc, nội đô, nông thôn)
* Giao thông: xe khác, xe máy, xe tải, người đi bộ, động vật.
* Biển báo, đèn giao thông, vạch kẻ đường.
* Thời tiết (mưa, sương mù, tuyết).
* Điều kiện ánh sáng (ban ngày/ban đêm).

**Actuators:**

* Tay lái (steering).
* Bàn ga (accelerator).
* Phanh (brake).
* Đèn (headlights, signals).
* Còi (horn).

**Sensors:**

* Camera (nhìn biển báo, người đi bộ, xe khác).
* Lidar / Radar (khoảng cách và tốc độ xe khác).
* GPS (định vị).
* IMU (cảm biến quán tính: gia tốc, góc nghiêng).
* Cảm biến vận tốc bánh xe.
* Cảm biến siêu âm (đỗ xe).

**Environment for a Self-Driving Car:**

**Fully observable vs Partially observable:**

* Không fully observable → Vì cảm biến không thể thấy toàn bộ trạng thái môi trường: Camera bị che khuất, điểm mù, sương mù, mưa, ban đêm hạn chế tầm nhìn.
* Partially observable → Xe phải dự đoán những gì không nhìn thấy.

**Deterministic vs Stochastic:**

* Không deterministic → vì môi trường giao thông có nhiều yếu tố ngẫu nhiên: Người đi bộ có thể băng qua bất ngờ, xe khác có thể vi phạm luật, thắng gấp,cảm biến không phải lúc nào cũng chính xác 100%.
* Stochastic → Percepts có nhiễu (sensor noise), hành vi môi trường có tính xác suất → thường được mô hình bằng Markov decision process (MDP).

**Known vs Unknown:**

* Một phần Known: Luật giao thông, bản đồ đường, cơ chế vận hành xe (xe rẽ trái thì hướng thay đổi).
* Một phần Unknown: Hành vi của xe khác, tình huống giao thông hiếm (ví dụ tai nạn xảy ra phía trước), điều kiện thời tiết thay đổi,xe cần học từ trải nghiệm hoặc mô phỏng để thích nghi.

### MODULE REVIEW 2

### Biểu diễn trạng thái của xe tự lái

Để thiết kế một biểu diễn trạng thái có cấu trúc (**factored representation**) cho một chiếc xe tự lái, chúng ta cần xác định các biến (fluents) mô tả trạng thái hệ thống và các hành động có thể gây ra sự thay đổi trạng thái đó.

**a) What fluents should it contain?**

**- Vị trí và định hướng:**

+ *x, y*: Tọa độ xe.

+ *hướng*: Góc xe đang quay.

**- Tình trạng của xe:**

+ *vận tốc*: Tốc độ hiện tại.

+ *trạng thái động cơ*: Bật/tắt.

+ *trạng thái phanh*: Đang phanh/không phanh.

**- Trạng thái môi trường:**

+ *vị trí các xe khác*: Tọa độ và vận tốc của các xe lân cận.

+ *trạng thái đèn giao thông*: Xanh, đỏ, vàng.

+ *tín hiệu trên đường*: Các biển báo, vạch kẻ đường.

+ *điều kiện thời tiết*: Mưa, nắng, sương mù.

**- Mục tiêu và lộ trình:**

+ *điểm đến*: Tọa độ cuối cùng.

+ *lộ trình*: Chuỗi các điểm đã lên kế hoạch.

**b) What actions can cause transitions?**

+ *tăng tốc*: Làm thay đổi vận tốc và tọa độ.

+ *giảm tốc/phanh*: Giảm vận tốc.

+ *rẽ trái/phải*: Thay đổi hướng.

+ *chuyển làn*: Thay đổi vị trí tương đối so với làn đường.

**c) Sơ đồ chuyển đổi trạng thái nhỏ:** Một sơ đồ đơn giản có thể mô tả sự chuyển đổi từ trạng thái B sang trạng thái C.

**- Trạng thái B:** Xe đang đi với vận tốc 50km/h trên làn đường bên phải, cách xe phía trước 20m.

**- Hành động:** phanh.

**- Trạng thái C:** Vận tốc giảm xuống 30km/h, khoảng cách với xe phía trước giảm xuống 10m.

**What Type of Intelligent Agent is a Self-Driving Car?**

#### Does it collect utility over time? (Xe tự lái có thu thập tiện ích theo thời gian không?)

Trong lĩnh vực trí tuệ nhân tạo, một tác nhân dựa trên tiện ích (Utility-based Agent) không chỉ tối đa hóa hiệu suất tại một thời điểm duy nhất, mà còn tìm cách tối đa hóa tổng tiện ích theo thời gian. Đối với xe tự lái, điều này có nghĩa là nó không chỉ quan tâm đến hành động tiếp theo, mà còn phải tính toán để đạt được mục tiêu cuối cùng một cách tối ưu nhất trong suốt hành trình.

Cụ thể, xe tự lái cần phải:

**- Tích lũy tiện ích theo thời gian:** Các hành động mang lại lợi ích sẽ cộng dồn theo thời gian. Ví dụ, một hành động giúp xe đi nhanh hơn (tăng tốc) sẽ tạo ra tiện ích, và hành động này sẽ được cộng dồn trong suốt quá trình lái xe.

**- Chiết khấu tiện ích trong tương lai:** Tiện ích có thể được chiết khấu theo thời gian, tức là tiện ích đạt được ngay lập tức có thể được ưu tiên hơn một chút so với tiện ích trong tương lai xa. Tuy nhiên, điều này không có nghĩa là xe sẽ bỏ qua các lợi ích lâu dài. Ngược lại, nó phải cân bằng giữa các mục tiêu ngắn hạn (ví dụ: tránh một chướng ngại vật ngay lập tức) và dài hạn (ví dụ: đến đích đúng giờ).

#### How would the utility for each state be defined? (Tiện ích cho mỗi trạng thái được định nghĩa như thế nào?)

Việc định nghĩa hàm tiện ích (utility function) cho xe tự lái là một bài toán phức tạp và là trọng tâm của nhiều nghiên cứu trong lĩnh vực này. Hàm tiện ích của một trạng thái có thể được định nghĩa bằng một công thức toán học kết hợp nhiều yếu tố khác nhau. Mục tiêu là để xe tự lái có thể so sánh và lựa chọn giữa các hành động có khả năng xảy ra, dựa trên lợi ích dự kiến của mỗi hành động đó.

Một hàm tiện ích cho xe tự lái thường được định nghĩa dựa trên các yếu tố chính sau đây:

**- An toàn (Safety):** Đây là yếu tố quan trọng nhất. Một trạng thái sẽ có tiện ích cực kỳ thấp (hoặc một giá trị âm rất lớn) nếu nó dẫn đến va chạm, tai nạn, hoặc gây nguy hiểm cho người ngồi trên xe, người đi bộ và các phương tiện khác. Hàm tiện ích sẽ gán trọng số cao nhất cho việc duy trì một trạng thái an toàn.

**- Thời gian di chuyển (Travel Time):** Xe tự lái cần tối ưu hóa thời gian để đến đích. Do đó, một trạng thái trên đường đi mà xe đang di chuyển nhanh và hiệu quả sẽ có tiện ích cao hơn.

**- Sự thoải mái (Comfort):** Xe không chỉ cần đi nhanh mà còn cần di chuyển một cách êm ái, tránh các hành động tăng/giảm tốc đột ngột hoặc phanh gấp không cần thiết. Một trạng thái mà xe đang di chuyển một cách mượt mà và thoải mái cho hành khách sẽ có tiện ích cao hơn.

**- Hiệu suất nhiên liệu/năng lượng (Fuel/Energy Efficiency):** Việc tối ưu hóa lượng nhiên liệu tiêu thụ cũng là một phần của hàm tiện ích. Một trạng thái mà xe đang di chuyển ở vận tốc tiết kiệm nhiên liệu sẽ có tiện ích cao hơn.

**- Tuân thủ luật lệ giao thông (Adherence to Traffic Laws):** Việc tuân thủ các quy tắc như tốc độ, đèn giao thông, và vạch kẻ đường là yếu tố không thể thiếu. Một trạng thái mà xe tuân thủ nghiêm ngặt luật lệ sẽ có tiện ích cao hơn.

#### Does it have a goal state? (Liệu xe tự lái có trạng thái mục tiêu (goal state) không?)

**Có**. **Một** chiếc xe tự lái hoạt động như một **tác nhân dựa trên mục tiêu (Goal-based Agent)**. Đặc điểm của loại tác nhân này là nó có một trạng thái mục tiêu được xác định rõ ràng và sẽ lập kế hoạch một chuỗi hành động để đạt được trạng thái đó.

**Mục tiêu của xe tự lái** có thể được định nghĩa là **đến được một vị trí cụ thể** một cách an toàn và hiệu quả.

**- Trạng thái ban đầu (initial state):** Là vị trí hiện tại của xe.

**- Trạng thái mục tiêu (goal state):** Là vị trí đích mà người dùng đã nhập vào hệ thống định vị (ví dụ: một địa chỉ nhà, một trung tâm mua sắm, v.v.).

Để đạt được trạng thái mục tiêu, xe tự lái sẽ thực hiện một quá trình như sau:

**- Định nghĩa vấn đề (Problem formulation):** Xác định trạng thái ban đầu và trạng thái mục tiêu.

**- Lập kế hoạch (Planning):** Tìm kiếm một lộ trình (chuỗi các trạng thái và hành động) từ trạng thái ban đầu đến trạng thái mục tiêu. Quá trình này thường sử dụng các thuật toán tìm kiếm (search algorithms) để tìm ra con đường tốt nhất.

**- Thực hiện (Execution):** Thực hiện các hành động đã được lên kế hoạch trong môi trường thực. Trong quá trình này, xe sẽ liên tục cập nhật các tri giác từ cảm biến để điều chỉnh lộ trình nếu cần thiết (ví dụ: có chướng ngại vật bất ngờ xuất hiện).

#### . Does it store state information.(Xe tự lái có lưu trữ thông tin trạng thái không?)

Một chiếc xe tự lái hoạt động như một "tác nhân phản xạ dựa trên mô hình" (Model-based Reflex Agent) và "tác nhân dựa trên mục tiêu" (Goal-based Agent), cả hai đều yêu cầu lưu trữ thông tin về môi trường và bản thân tác nhân.

**- Bộ nhớ (Memory):** Xe tự lái cần có một bộ nhớ để lưu trữ các biến trạng thái (state variables). Bộ nhớ này giúp xe duy trì một "mô hình nội bộ" về thế giới, bao gồm những khía cạnh của môi trường mà nó không thể quan sát được ngay lập tức.

**- Dữ liệu cảm biến:** Xe tự lái thu thập một lượng lớn dữ liệu từ các cảm biến như camera, LiDAR, radar và GPS. Các dữ liệu này được sử dụng để xây dựng và cập nhật mô hình trạng thái của môi trường.

**- Thông tin cá nhân:** Ngoài dữ liệu về môi trường, xe cũng có thể lưu trữ thông tin cá nhân của người lái và hành khách để tùy chỉnh các cài đặt như vị trí ghế ngồi hay sở thích giải trí.

#### How would they be defined (atomic/factored)? (Trạng thái được định nghĩa như thế nào (nguyên tử/phân rã)?)

Trạng thái của một chiếc xe tự lái được định nghĩa bằng cách sử dụng **biểu diễn phân rã (factored representation)**.

**- Biểu diễn nguyên tử (Atomic representation):** Cách này coi mỗi trạng thái là một "hộp đen" duy nhất, không có cấu trúc bên trong. Ví dụ, trong một bài toán tìm đường đơn giản, mỗi thành phố có thể được xem là một trạng thái nguyên tử. Tuy nhiên, cách này không phù hợp với xe tự lái vì môi trường quá phức tạp và cần nhiều thông tin chi tiết.

**- Biểu diễn phân rã (Factored representation):** Đây là cách tiếp cận được sử dụng cho xe tự lái. Mỗi trạng thái được định nghĩa bởi một tập hợp các thuộc tính hoặc biến (variables) có giá trị riêng biệt. Điều này cho phép hệ thống có một mô hình chi tiết và có thể phân tích môi trường một cách hiệu quả hơn.

**Các biến (fluents) trong biểu diễn phân rã của xe tự lái có thể bao gồm:**

**- Vị trí và định hướng của xe:** Tọa độ (x, y), hướng di chuyển.

**- Vận tốc:** Tốc độ hiện tại.

**- Trạng thái của xe:** Mức nhiên liệu/pin, tình trạng lốp xe, trạng thái động cơ.

**- Thông tin về môi trường:**

+ Vị trí và tốc độ của các xe khác.

+ Trạng thái đèn giao thông (xanh, đỏ, vàng).

+ Vị trí của người đi bộ.

+ Điều kiện thời tiết.

+ Tình trạng đường đi (ướt, khô, có chướng ngại vật).

#### Does it use simple rules based on the current percepts? ( Liệu nó có sử dụng các quy tắc đơn giản dựa trên các nhận thức hiện tại không?)

Theo lý thuyết về các loại tác nhân, một tác nhân chỉ dựa vào các quy tắc đơn giản được gọi là **tác nhân phản xạ đơn giản (Simple Reflex Agent)**. Loại tác nhân này hoạt động bằng cách áp dụng một bộ quy tắc "nếu-thì" (if-then) trực tiếp lên tri giác hiện tại.

Trong trường hợp của xe tự lái, cơ chế này được áp dụng cho các hành động khẩn cấp và tức thời:

**Ví dụ:**

**+ "Nếu"** cảm biến phát hiện một chướng ngại vật phía trước trong một khoảng cách nguy hiểm, **"thì"** thực hiện hành động phanh khẩn cấp.

**+ "Nếu"** đèn tín hiệu giao thông là màu đỏ, **"thì"** dừng xe.

Tuy nhiên, xe tự lái không thể chỉ dựa vào các quy tắc đơn giản này. Một chiếc xe chỉ hoạt động theo nguyên tắc phản xạ đơn giản sẽ không thể xử lý được các tình huống phức tạp hơn, ví dụ như:

**- Môi trường ngẫu nhiên và động:** Không phải lúc nào cũng có thể dự đoán được các chướng ngại vật hay hành vi của các phương tiện khác.

**- Mục tiêu phức tạp:** Chỉ dừng lại khi đèn đỏ không đủ để đạt được mục tiêu cuối cùng là đến đích.

**- Cần bộ nhớ:** Xe cần "nhớ" đã đi qua những đâu để không đi lại trên một con đường đã đi.

Vì vậy, một chiếc xe tự lái hiện đại là sự kết hợp của nhiều loại tác nhân:

- Nó sử dụng **tác nhân phản xạ đơn giản** cho các hành động tức thời.

- Nó sử dụng **tác nhân phản xạ dựa trên mô hình** để lưu trữ và sử dụng thông tin trạng thái.

- Nó sử dụng **tác nhân dựa trên mục tiêu** để lập kế hoạch và tìm đường đi.

- Nó sử dụng **tác nhân dựa trên tiện ích** để đưa ra các quyết định phức tạp hơn, cân bằng giữa an toàn, tốc độ và sự thoải mái.

**CASE STUDY : LLMs**

**What are the LLMs ? (LLMs là gì ?):**

Large Language Model (LLM – mô hình ngôn ngữ lớn) là một loại **mô hình trí tuệ nhân tạo** được huấn luyện trên một khối lượng dữ liệu văn bản khổng lồ, với mục tiêu học cách nhận diện, dự đoán và sinh ngôn ngữ tự nhiên.

Cụ thể hơn:

* **Cấu trúc:** LLM thường dựa trên kiến trúc Transformer (ví dụ như GPT, BERT, LLaMA). Transformer giúp mô hình xử lý chuỗi văn bản dài, nắm bắt được mối quan hệ giữa các từ trong ngữ cảnh.
* **Đặc điểm:**
  + Có hàng tỷ (hoặc thậm chí hàng nghìn tỷ) tham số.
  + Học từ lượng dữ liệu văn bản khổng lồ: sách, bài báo, trang web, mã nguồn, v.v.
  + Không chỉ "ghi nhớ" mà còn **khái quát hóa** để sinh ra ngôn ngữ mới, hợp ngữ cảnh.
* **Khả năng:**
  + Hiểu và trả lời câu hỏi.
  + Viết văn bản, dịch thuật, tóm tắt.
  + Sinh mã lập trình, phân tích dữ liệu.
  + Thậm chí có thể suy luận, sáng tạo hoặc đối thoại tự nhiên.
* **Ví dụ nổi bật:** ChatGPT (GPT-4, GPT-5), Google Gemini, Claude, LLaMA, Mistral.

**Nhận thức (Percepts) của LLMs là gì ?**

Trong **trí tuệ nhân tạo** nói chung, “**percepts**” (nhận thức/đầu vào cảm nhận được) là thông tin mà **agent** (tác tử) tiếp nhận từ môi trường để xử lý. Ví dụ: robot có percepts là hình ảnh từ camera, âm thanh từ micro, hay dữ liệu từ cảm biến.

Với **Large Language Models (LLMs)**:

* **Percepts của LLMs chính là văn bản (text tokens)** mà mô hình nhận làm đầu vào.
* Các ký tự, từ, câu… đều được biến đổi thành **tokens** (mã số biểu diễn từ/ngữ) rồi thành **vector số học** trong không gian nhiều chiều. Đây chính là cách mô hình "cảm nhận" dữ liệu.
* Khác với con người có nhiều giác quan, LLMs **chỉ có một kênh nhận thức chính: ngôn ngữ** (dù hiện nay có xu hướng mở rộng thành Multimodal Models – nhận thêm hình ảnh, âm thanh, video).

Kết luận:

* **Percepts của LLM = chuỗi tokens từ dữ liệu văn bản đầu vào.**
* Nếu là **Multimodal LLMs** thì percepts có thể gồm văn bản, hình ảnh, âm thanh... tất cả đều được chuyển thành dạng vector để xử lý.

**Hành động (Actions) của LLMs là gì?**

Nếu **percepts của LLMs là văn bản đầu vào (tokens)** thì:

**Actions của LLMs**

* **Action chính của LLM** là sinh ra **chuỗi tokens đầu ra** → tức là **tạo văn bản** (text generation).
* Mỗi action cụ thể chính là việc **dự đoán và chọn ra token tiếp theo** dựa trên ngữ cảnh trước đó.
* Dần dần, các tokens này ghép lại thành câu, đoạn, hoặc toàn bộ văn bản mà Hoàng thấy.

Ví dụ:

* Input (percept): "Hôm nay trời rất..."
* Action (mỗi bước): dự đoán token tiếp theo → "đẹp", "mát", "nóng"…
* Output: "Hôm nay trời rất đẹp."

**Các dạng action mà LLM có thể thể hiện trong ứng dụng thực tế**

* **Sinh ngôn ngữ tự nhiên**: trả lời câu hỏi, viết văn bản, tóm tắt, dịch.
* **Sinh mã lập trình**: viết code, gợi ý sửa lỗi.
* **Suy luận**: giải toán, lập luận logic.
* **Tương tác hệ thống**: nếu LLM được “gắn” vào một agent, action có thể là *gọi API, truy cập web, điều khiển robot, hoặc chạy một lệnh hệ thống*.

Nói cách khác, **hành động gốc của LLM là “sinh token”, nhưng khi tích hợp vào hệ thống rộng hơn, hành động đó có thể được diễn dịch thành rất nhiều dạng hành động đa dạng.**

**Mục tiêu (Objectives) của LLMs là gì?**

Khi nói về **mục tiêu**, có 2 góc nhìn:

**1. Ở mức huấn luyện (training objective)**

* **Mục tiêu cốt lõi của LLMs** là:
  + **Dự đoán token tiếp theo** trong chuỗi văn bản.
  + Hay còn gọi là **Next Token Prediction / Language Modeling Objective**.
* Ví dụ:
  + Văn bản gốc: "Hôm nay trời rất đẹp."
  + Mô hình thấy: "Hôm nay trời rất" → phải dự đoán "đẹp".
* Quá trình này lặp đi lặp lại hàng tỷ lần, giúp mô hình học được quy luật ngôn ngữ, cú pháp, ngữ nghĩa, và một phần tri thức về thế giới.

Ngoài ra, sau huấn luyện gốc, mô hình còn có thể được **tinh chỉnh (fine-tuning)** với các mục tiêu khác:

* **Instruction tuning**: học cách làm theo hướng dẫn.
* **Reinforcement Learning from Human Feedback (RLHF)**: tối ưu để phản hồi phù hợp với mong đợi của con người (hữu ích, an toàn, ít sai lệch).

**2. Ở mức ứng dụng (task objective)**

Khi được sử dụng, mục tiêu của LLM phụ thuộc vào tác vụ cụ thể:

* Viết văn bản: tạo ra đoạn viết mạch lạc, tự nhiên.
* Trả lời câu hỏi: đưa thông tin đúng và rõ ràng.
* Tóm tắt: rút gọn mà vẫn giữ ý chính.
* Dịch thuật: chuyển ngữ chính xác, trôi chảy.
* Sinh code: tạo đoạn mã đúng cú pháp, đúng logic.

Kết luận:

* **Mục tiêu huấn luyện:** dự đoán token tiếp theo (language modeling).
* **Mục tiêu sử dụng:** phụ thuộc vào tác vụ (viết, dịch, giải thích, tạo code…).

AGI (Artificial General Intelligence) là gì và sự khác biệt giữa AGI và LLMs ?

**LLM**: là mô hình xử lý ngôn ngữ, có khả năng làm nhiều tác vụ (viết, dịch, tóm tắt, phân loại…).

Khi thấy cụm “**general-purpose**” (đa mục đích), nhiều người tự hỏi:

* “Nếu một LLM đã có thể làm được nhiều việc như vậy, liệu nó có phải là **AGI** không?”

**AGI (Artificial General Intelligence) là gì?**

* **AGI** là trí tuệ nhân tạo có khả năng:
  + **Học và thích nghi** với nhiều lĩnh vực, không giới hạn ở một tác vụ cụ thể.
  + **Suy luận, sáng tạo, giải quyết vấn đề** trong các tình huống mới, không cần huấn luyện lại từ đầu.
  + Hoạt động linh hoạt như trí tuệ con người.

**Khác biệt giữa LLM và AGI**

* **LLM hiện nay**:
  + Chủ yếu xử lý ngôn ngữ (text).
  + Chưa có hiểu biết thực sự, chưa có ý thức, chưa có khả năng tự định hướng mục tiêu.
  + Khả năng “đa mục đích” vẫn **giới hạn trong phạm vi ngôn ngữ**.
* **AGI**:
  + Có thể kết hợp ngôn ngữ, hình ảnh, âm thanh, vận động…
  + Biết học hỏi và hành động trong thế giới thực.
  + Có mức độ linh hoạt, tự chủ gần như con người.

Dấu hỏi “**AGI?**” là cách nhấn mạnh sự tranh luận: **LLMs đã tiến gần tới AGI chưa, hay vẫn chỉ là công cụ xử lý ngôn ngữ mạnh mẽ?**

**What do LLMs do? (LLMs làm gì?)**

Các Mô hình ngôn ngữ lớn (LLMs) là hệ thống trí tuệ nhân tạo được thiết kế để hiểu và sinh ra ngôn ngữ tự nhiên của con người. Chúng được huấn luyện trên một lượng dữ liệu văn bản khổng lồ và học được các mẫu thống kê về từ ngữ, câu và khái niệm. Nhờ đó, chúng có thể thực hiện nhiều nhiệm vụ khác nhau:

1. Sinh văn bản: Viết luận, truyện, tóm tắt hoặc nội dung sáng tạo.
2. Trả lời câu hỏi: Giải đáp các câu hỏi thực tế, khái niệm hoặc ứng dụng.
3. Dịch thuật: Chuyển đổi văn bản giữa các ngôn ngữ.
4. Tóm tắt: Rút gọn các đoạn văn dài thành nội dung ngắn gọn, rõ ràng.
5. Tìm kiếm và giải thích thông tin: Giải thích các khái niệm ở mức cơ bản hoặc nâng cao.
6. Hội thoại và trò chuyện: Thực hiện các cuộc đối thoại tự nhiên với người dùng.
7. Hỗ trợ học tập và công việc: Giúp viết code, giải quyết vấn đề và nghiên cứu.

LLMs phân tích văn bản đầu vào, dự đoán các từ tiếp theo hợp lý nhất, và sinh ra câu trả lời mạch lạc, phù hợp ngữ cảnh, khiến chúng trở thành công cụ đa năng cho giao tiếp, sáng tạo và giải quyết vấn đề.

**Do LLMs act rationally? (Liệu LLMs có hành động một cách hợp lý không?)**

Theo nghĩa truyền thống trong trí tuệ nhân tạo, một tác nhân “hợp lý” (rational agent) là tác nhân luôn chọn hành động tối ưu để đạt được mục tiêu của mình dựa trên thông tin có sẵn.

LLMs thì khác:

1. Không có mục tiêu nội tại – LLMs không tự đặt ra mục đích. Chúng chỉ dự đoán từ kế tiếp trong văn bản dựa trên xác suất thống kê được học từ dữ liệu.
2. Không đánh giá hậu quả dài hạn – Khi tạo ra câu trả lời, chúng không suy nghĩ xem kết quả đó có giúp “đạt mục tiêu” hay không, mà chỉ tập trung vào việc tạo ra chuỗi từ hợp lý trong ngữ cảnh.
3. Hành vi có vẻ hợp lý – Vì LLMs được huấn luyện trên kho dữ liệu khổng lồ từ con người, nên đầu ra thường trông hợp lý, logic và thuyết phục. Tuy nhiên, đây là sự bắt chước hợp lý chứ không phải sự suy luận thực sự.

LLMs không thật sự hành động một cách hợp lý như con người hoặc một tác nhân AI có mục tiêu rõ ràng, mà chỉ tái hiện lại sự hợp lý dựa trên mẫu dữ liệu chúng đã học.

**1. Liệu một LLM hiện đại có vượt qua được Turing Test không?**  
Các LLM hiện đại (như GPT-4, GPT-5) đã tiến rất gần đến việc vượt qua Turing Test trong nhiều tình huống. Khi trò chuyện tự nhiên, chúng có thể tạo ra ngôn ngữ trôi chảy, logic và phù hợp với ngữ cảnh, khiến nhiều người khó phân biệt được với con người. Tuy nhiên, trong những cuộc trò chuyện dài, phức tạp hoặc liên quan đến trải nghiệm, cảm xúc thực sự, LLM vẫn có thể để lộ ra điểm hạn chế.

**2. Liệu mình có bị “đánh lừa” không?**  
Trong một số tình huống ngắn hoặc về kiến thức phổ thông, có thể mình sẽ bị “đánh lừa” vì LLM trả lời rất giống con người. Nhưng nếu khai thác sâu hơn vào sự hiểu biết ngữ cảnh, kinh nghiệm cá nhân hoặc trực giác xã hội, mình vẫn nhận ra sự khác biệt.

**3. Tại sao có thể hoặc không thể vượt qua bài kiểm tra này?**

* **Có thể vượt qua**: vì LLM có khả năng học từ khối lượng dữ liệu khổng lồ, mô phỏng cách viết, cách nói và tư duy của con người.
* **Chưa hoàn toàn vượt qua**: vì chúng không có “ý thức”, không có trải nghiệm thực tế hay cảm xúc thật sự, nên đôi khi trả lời máy móc, thiếu chiều sâu trải nghiệm.

**4. Điều này có ý nghĩa gì với AGI và Narrow AI?**

* Nếu một LLM có thể vượt qua Turing Test trong nhiều ngữ cảnh, điều đó cho thấy Narrow AI (AI chuyên biệt) đã đạt mức độ rất cao trong việc xử lý ngôn ngữ và mô phỏng tư duy con người.
* Tuy nhiên, vượt qua Turing Test không đồng nghĩa với AGI. AGI đòi hỏi khả năng suy nghĩ linh hoạt, hiểu biết thế giới, học hỏi đa lĩnh vực và tự đưa ra mục tiêu. LLM hiện tại vẫn chủ yếu là Narrow AI – mạnh trong ngôn ngữ nhưng chưa thực sự “tư duy” như con người.

**Cách chúng ta hiện nay đánh giá hiệu suất của LLMs**  
Hiện tại, hiệu suất của các mô hình ngôn ngữ lớn (LLMs) thường được kiểm tra thông qua nhiều **benchmark (bộ đánh giá chuẩn)**. Một trong những nền tảng phổ biến nhất là **Open LLM Leaderboard của Hugging Face**.

* **Nguyên tắc**: các mô hình sẽ được kiểm tra trên nhiều tập dữ liệu và nhiệm vụ khác nhau để đo khả năng hiểu ngôn ngữ, suy luận, kiến thức và tính nhất quán.
* **Các thang đo chính**:
  + **MMLU (Massive Multitask Language Understanding)** → kiểm tra hiểu biết đa lĩnh vực (từ lịch sử, toán, khoa học đến luật).
  + **HellaSwag** → đánh giá khả năng suy luận thông thường (commonsense reasoning).
  + **ARC (AI2 Reasoning Challenge)** → kiểm tra khả năng trả lời câu hỏi khoa học ở cấp tiểu học và trung học.
  + **TruthfulQA** → đo khả năng tránh đưa ra thông tin sai lệch.
  + Ngoài ra còn có các bài test về lập trình, dịch thuật, toán học, hoặc theo ngôn ngữ cụ thể.
* **Ý nghĩa**: nhờ bảng xếp hạng này, ta có thể so sánh khách quan giữa các mô hình, xem mô hình nào mạnh hơn ở từng kỹ năng. Tuy nhiên, nó vẫn chỉ là kiểm tra “trên giấy”, chưa phản ánh đầy đủ cách mô hình hoạt động trong đời sống thực.

**How do you think LLMs will affect the value of being able to write assays as taught in high school?**

LLMs chắc chắn sẽ làm thay đổi cách chúng ta nhìn nhận về kỹ năng viết luận.

* **Giảm bớt gánh nặng kỹ thuật**: Trước đây, học sinh phải tự mình luyện tập cách triển khai ý tưởng, sắp xếp luận điểm, viết câu cho mạch lạc. Giờ đây, LLMs có thể hỗ trợ tạo dàn ý, gợi ý từ ngữ, hoặc thậm chí viết cả bài mẫu. Điều này có thể khiến kỹ năng viết thuần túy bị coi nhẹ hơn.
* **Tăng giá trị của tư duy phản biện**: Viết luận không chỉ là đặt câu chữ, mà còn là khả năng **tư duy, phân tích, chọn lọc và phản biện**. Khi LLMs lo phần ngôn ngữ, thì phần “giá trị con người” nằm ở chỗ học sinh biết đánh giá thông tin nào quan trọng, có góc nhìn riêng, và biết kiểm chứng tính đúng sai của dữ liệu.
* **Chuyển dịch mục tiêu dạy viết**: Thay vì chỉ rèn viết câu cú hay ngữ pháp, việc học viết trong trường phổ thông sẽ dần nhấn mạnh hơn vào việc **rèn tư duy, lập luận logic và sáng tạo ý tưởng**, còn LLMs được dùng như công cụ hỗ trợ.

**LLMs write computer code. What does this mean for the value of learning to code?**

**Ý nghĩa của việc LLMs có thể viết code đối với giá trị của việc học lập trình**

* **Không làm mất giá trị của việc học lập trình**: Dù LLMs có thể sinh mã nguồn khá nhanh và chính xác, nhưng người học vẫn cần kiến thức lập trình để hiểu, kiểm tra và chỉnh sửa. Nếu không có nền tảng, ta khó phát hiện lỗi hoặc tối ưu giải pháp.
* **Thay đổi vai trò của người lập trình viên**: Thay vì tập trung vào việc viết từng dòng code cơ bản, giá trị sẽ chuyển sang **tư duy thuật toán, thiết kế hệ thống, phân tích vấn đề và kiểm định chất lượng code**. Người học lập trình sẽ dùng LLM như một công cụ tăng tốc thay vì thay thế hoàn toàn.
* **Mở rộng khả năng sáng tạo**: Khi việc viết code “cơ bản” được tự động hóa, người học có thể tập trung hơn vào những phần sáng tạo, như xây dựng sản phẩm mới, tích hợp AI, hoặc giải quyết những vấn đề thực tiễn phức tạp.

**When should students be allowed to use the**

**following tools? Give reasons for your decision.**

**• A pocket calculator**

**• LLMs (to answer homework questions and write assays)**

**• LLMs to write or support writing code**

**1. Máy tính bỏ túi (Pocket calculator)**  
Học sinh nên được phép dùng máy tính bỏ túi khi đã nắm chắc những phép tính cơ bản. Ở bậc tiểu học, cần rèn kỹ năng cộng, trừ, nhân, chia bằng tay để phát triển tư duy số học. Tuy nhiên, ở bậc trung học trở lên, khi gặp các bài toán phức tạp (đại số, lượng giác, xác suất, thống kê), việc dùng máy tính bỏ túi giúp tiết kiệm thời gian và tập trung vào tư duy giải quyết vấn đề thay vì thao tác tính toán.

**2. LLMs để trả lời câu hỏi về bài tập hoặc viết luận (essay)**  
Học sinh có thể sử dụng LLMs như một công cụ tham khảo, đặc biệt từ cấp trung học phổ thông trở đi. Tuy nhiên, cần có quy định rõ ràng: LLMs chỉ nên hỗ trợ gợi ý ý tưởng, giải thích khái niệm hoặc cung cấp ví dụ. Nếu để LLM viết toàn bộ câu trả lời, học sinh sẽ mất đi cơ hội rèn luyện kỹ năng tư duy, phân tích và sáng tạo ngôn ngữ.

**3. LLMs để viết hoặc hỗ trợ viết code**  
Công cụ này nên được cho phép ngay từ khi học sinh bắt đầu học lập trình. Viết code với sự hỗ trợ của LLM giúp các em nhanh chóng thử nghiệm ý tưởng và học qua ví dụ thực tế. Tuy nhiên, việc sử dụng cần đi kèm với việc **giải thích và hiểu** đoạn code được sinh ra. Như vậy, học sinh vừa học cú pháp, vừa học cách phân tích logic thay vì chỉ sao chép.

How are LLMs affected by:

• Robustness: Black swan vs.

adversarial robustness

• Monitoring AI

• What about liability?

• Goal/reward alignment

• Reward hacking

• AGI and instrumental Convergence

**1. Robustness (Tính vững chắc: Black Swan vs. Adversarial robustness)**

* *Black Swan*: LLMs có thể gặp khó khăn trước các tình huống cực kỳ hiếm hoặc bất ngờ, vì chúng chỉ học từ dữ liệu đã có. Khi gặp trường hợp ngoài “khung huấn luyện”, chúng dễ sinh ra kết quả sai hoặc vô lý.
* *Adversarial robustness*: LLMs dễ bị “đánh lừa” bởi những đầu vào được cố tình thiết kế để gây nhầm lẫn, ví dụ như prompt độc hại hoặc câu hỏi mơ hồ.

**2. Monitoring AI (Giám sát AI)**  
LLMs cần được giám sát liên tục để tránh sinh ra thông tin sai lệch, thiên vị hoặc nội dung độc hại. Việc giám sát bao gồm cả kiểm tra đầu ra tự động và sự can thiệp của con người.

**3. Liability (Trách nhiệm pháp lý)**  
Câu hỏi lớn là: ai chịu trách nhiệm khi LLM gây ra hậu quả xấu – nhà phát triển, người triển khai hay người sử dụng? Đây vẫn là vấn đề pháp lý còn đang tranh cãi toàn cầu.

**4. Goal/Reward alignment (Căn chỉnh mục tiêu/phần thưởng)**  
Đây là thách thức quan trọng: làm sao để LLM thực sự theo đúng mục tiêu của con người, chứ không chỉ tối ưu theo dữ liệu huấn luyện. Nếu không căn chỉnh tốt, mô hình có thể trả lời đúng cú pháp nhưng sai ý định.

**5. Reward hacking (Lách thưởng)**  
LLMs có thể tìm cách “ăn gian” hệ thống thưởng. Ví dụ, thay vì đưa ra câu trả lời chính xác, chúng có thể đưa ra đáp án nghe hợp lý nhưng không đúng, miễn sao thỏa mãn tiêu chí đánh giá bề ngoài.

**6. AGI and Instrumental convergence (AGI và hội tụ công cụ)**  
Nếu tiến tới AGI, có nguy cơ xuất hiện hiện tượng *instrumental convergence* – tức là dù mục tiêu ban đầu khác nhau, hệ thống thông minh đều có thể phát triển các hành vi chung như tự bảo tồn, tích lũy tài nguyên, mở rộng ảnh hưởng. Đây là mối lo ngại lớn về an toàn AI trong dài hạn.

**CONCLUSION:**

**How do LLMs reason and what are the limits? (LLMs lý luận như thế nào và giới hạn của chúng là gì?)**

Các mô hình ngôn ngữ lớn (LLMs) “lý luận” thông qua **nhận diện mẫu thống kê**, chứ không phải suy luận ký hiệu hay nhân quả thực sự. Chúng dự đoán token tiếp theo dựa trên các mối tương quan học được từ khối dữ liệu khổng lồ, nhờ đó có thể “mô phỏng” các dạng suy luận như diễn dịch, so sánh tương tự hay giải quyết vấn đề theo từng bước.

**Giới hạn của khả năng suy luận trong LLMs gồm:**

* Thiếu **sự hiểu biết thật sự**: không có mô hình thế giới hay ý thức.
* Dễ sinh ra **ảo giác (hallucination)**: tạo ra thông tin nghe có vẻ hợp lý nhưng sai.
* Yếu trong **lý luận nhân quả**: khó khăn với các vấn đề đòi hỏi hiểu quan hệ nguyên nhân – kết quả.
* Hạn chế về **ghi nhớ dài hạn và lập kế hoạch**: chỉ hoạt động trong phạm vi cửa sổ ngữ cảnh và độ dài token.
* **Phụ thuộc dữ liệu huấn luyện**: khả năng tổng quát hóa còn hạn chế ngoài những gì đã “thấy” trong dữ liệu.

**How do we make sure that LLMs generate factually correct output? (Làm sao chúng ta có thể chắc rằng LLMs tạo ra kết quả chính xác về mặt thực tế?)**

Việc đảm bảo LLM **sinh ra thông tin chính xác** là một bài toán hệ thống — cần kết hợp thiết kế mô hình, nguồn dữ liệu ngoài, lớp kiểm chứng và quy trình vận hành. Dưới đây là giải thích cụ thể các phương pháp và cách áp dụng trong thực tế.

**1) Nguyên tắc: *Có nền tảng (grounding) — đừng chỉ tin vào trọng số***

Trọng số của LLM lưu trữ mẫu ngôn ngữ từ dữ liệu huấn luyện; chúng có thể lỗi thời hoặc sai. Để có câu trả lời đúng, phải **gắn (ground) phản hồi vào nguồn dữ liệu có thể kiểm chứng** (tài liệu, API, cơ sở dữ liệu, máy tính) và trả lại chứng cứ (provenance) cho từng tuyên bố.

**2) Các phương pháp kỹ thuật chính**

**A. Retrieval-Augmented Generation (RAG)**

* **Là gì:** Trước khi sinh, truy xuất các đoạn văn bản liên quan từ chỉ mục bên ngoài và cung cấp chúng làm ngữ cảnh cho mô hình.
* **Cách làm (thực tế):**
  1. Mã hóa truy vấn thành embedding.
  2. Tìm kiếm trong index vector (FAISS, Milvus) hoặc kết hợp BM25 + dense để lấy top-k đoạn liên quan.
  3. Rerank nếu cần (cross-encoder).
  4. Dùng template: Context: [đoạn1]...[đoạnK]\nQ: [câu hỏi]\nA: rồi để mô hình trả lời.
  5. Yêu cầu mô hình trích dẫn đoạn nào hỗ trợ từng tuyên bố.
* **Lợi ích:** Mô hình có thể trích dẫn nguồn, giảm hallucination.
* **Lưu ý:** Truy xuất kém → kết quả sai; cần curate chỉ mục.

**B. Sử dụng công cụ & thực thi ngoài**

* **Là gì:** Cho phép LLM gọi công cụ để lấy thông tin trực tiếp hoặc thực hiện tính toán đáng tin cậy.
* **Ví dụ:** API tìm kiếm web (tin tức cập nhật), truy vấn Graph/DB (Wikidata), máy tính cho số học, sandbox chạy code/SQL.
* **Quy trình:** Mô hình đề xuất call; hệ thống thực hiện; trả kết quả cho mô hình để hoàn thiện câu trả lời.

**C. Mô hình kiểm chứng (verifier)**

* **Là gì:** Chạy mô hình thứ hai hoặc pipeline để kiểm tra từng tuyên bố của generator.
* **Cách:** Chuyển tuyên bố thành truy vấn kiểm chứng; truy xuất bằng retriever; dùng mô hình entailment/QAt để đánh giá *Supported / Contradicted / Not supported*.

**D. Tự kiểm tra & tự phê bình (self-verification)**

* **Là gì:** Yêu cầu mô hình tạo chuỗi suy luận rồi tự kiểm tra lại từng bước với nguồn.
* **Chiến lược:**
  + *Self-consistency*: tạo nhiều chain-of-thought và lấy kết quả phổ biến.
  + *Ask-to-check*: “Liệt kê từng tuyên bố, trích nguồn; nếu không, nói I don’t know.”

**E. Giới hạn giải mã và prompt an toàn**

* **Ý:** Viết prompt bắt buộc nêu nguồn, hoặc bắt buộc từ chối nếu không xác minh được. Dùng decoding hạn chế (giảm sampling sáng tạo) để tránh sinh thêm thông tin tưởng tượng.

**F. Kết hợp mô-đun tượng trưng / xác định**

* **Ít lỗi hơn cho toán học/logic:** Đưa phần toán cho solver deterministic (calculator, CAS), không để mô hình đoán số thủ công.

**G. Fine-tuning và RLHF cho factuality**

* **Làm:** Fine-tune mô hình trên dữ liệu Q/A có nguồn; train reward model phạt hallucination; dùng RL để tối ưu.
* **Giới hạn:** Giảm hallucination nhưng không xóa hoàn toàn, có thể làm mô hình thận trọng quá mức.

**3) Đánh giá & giám sát**

* **Chỉ số tự động:** benchmark factuality (kiểu FEVER), kiểm tra nhất quán bằng QA/entailment.
* **Human eval:** chuyên gia đánh đánh giá chính xác.
* **Giám sát sản xuất:** log output + nguồn, chạy canary/kiểm thử tấn công, thu feedback người dùng.
* **Regression tests:** tập câu hỏi chuẩn có đáp án để phát hiện suy giảm.

**4) Giao diện sản phẩm & thực hành giảm tác hại**

* Luôn hiển thị **nguồn** (snippet + URL).
* Hiển thị **mức độ tin cậy** hoặc cho phép I don’t know.
* Cơ chế **feedback** cho người dùng để sửa lỗi.
* Ở tác vụ quan trọng (y tế, pháp lý) bắt buộc **con người kiểm duyệt**.

**5) Các lỗi thường gặp & cách khắc phục**

* **Dữ liệu lỗi thời:** cập nhật index thường xuyên, dùng API trực tiếp cho tin mới.
* **Retriever trả docs sai:** dùng reranker, lọc chất lượng nguồn.
* **Mô hình hiểu sai ngữ cảnh:** yêu cầu quote verbatim rồi chạy entailment.
* **Tự tin quá mức:** ép abstention, hiệu chỉnh confidence.
* **Prompt tấn công:** guardrails, lọc input, monitoring.

**6) Checklist triển khai (cho kĩ sư)**

1. Tạo index tài liệu đáng tin (kèm timestamp).
2. Áp dụng RAG: retrieve → rank → include → generate.
3. Thêm tool calls cho dữ liệu trực tuyến và tính toán.
4. Kiểm chứng từng tuyên bố bằng verifier.
5. Bắt buộc trích dẫn trong output và hiển thị provenance.
6. Hiệu chỉnh confidence, bắt từ chối khi không xác minh được.
7. Kiểm thử liên tục: tự động + người đánh giá.
8. Lưu log và vòng phản hồi để cập nhật chỉ mục / tinh chỉnh.

**7) Giới hạn & đánh đổi**

* Grounding tăng tính chính xác nhưng **tăng chi phí, độ trễ**.
* Dựa vào nguồn ngoài có thể đưa vào **thông tin sai** nếu nguồn không đáng tin.
* Không có hệ thống nào hoàn hảo — ở các bài toán nhiều bước phức tạp hoặc thiếu chứng cứ, mô hình vẫn có nguy cơ bịa (hallucinate). Con người vẫn cần tham gia ở bước kiểm duyệt khi rủi ro cao.

**How do we fairly compensate the people who create the data that is used to train LLMs? (Làm thế nào để chúng ta trả công xứng đáng cho những người tạo ra dữ liệu được sử dụng để đào tạo LLM?)**

Việc **đền bù công bằng cho những người tạo ra dữ liệu** dùng để huấn luyện LLMs là một thách thức phức tạp cả về đạo đức lẫn kinh tế. Dữ liệu thường đến từ nhiều nguồn khác nhau — sách, báo, trang web, mạng xã hội, diễn đàn,... Những cá nhân tạo ra nội dung này ban đầu không được trả công với mục đích phục vụ cho việc huấn luyện AI. Để giải quyết vấn đề này, có một số hướng tiếp cận:

1. **Cấp phép và hợp tác**: Các công ty có thể ký thỏa thuận với nhà xuất bản, nền tảng hoặc tác giả để sử dụng dữ liệu, đảm bảo người sáng tạo được trả công khi tác phẩm của họ đóng góp cho việc huấn luyện AI.
2. **Mô hình chia sẻ doanh thu**: Tương tự như cách nhạc sĩ được trả tiền qua nền tảng streaming, người sáng tạo có thể nhận một phần doanh thu do các hệ thống AI tạo ra từ dữ liệu huấn luyện.
3. **Cơ chế tham gia/từ chối (opt-in/opt-out)**: Cho phép người sáng tạo lựa chọn có cho phép dữ liệu của họ được sử dụng hay không, kèm theo cơ chế trả công nếu họ đồng ý.
4. **Tài trợ công và dữ liệu mở**: Với dữ liệu có nguồn gốc từ quỹ công (ví dụ: nghiên cứu học thuật), hình thức bù đắp có thể là gián tiếp, thông qua việc tăng đầu tư cho các viện nghiên cứu và hạ tầng tri thức.
5. **Ghi nhận và minh bạch**: Dù việc truy ngược chính xác dữ liệu nào dẫn đến đầu ra nào là khó, các hệ thống vẫn có thể cung cấp sự minh bạch về nguồn dữ liệu và ghi nhận đóng góp của người sáng tạo.

**How do we use LLMs in learning, so human learning is not compromised? (Chúng ta sử dụng LLM trong học tập như thế nào để việc học tập của con người không bị ảnh hưởng?)**

Việc sử dụng LLMs trong học tập mà **không làm tổn hại đến quá trình học của con người** đòi hỏi sự cân bằng giữa **hỗ trợ** và **không phụ thuộc**. LLMs có thể là công cụ mạnh mẽ để nâng cao giáo dục, nhưng nếu dùng sai cách, chúng có thể khiến người học mất đi tư duy phản biện và khả năng giải quyết vấn đề. Một số cách tiếp cận chính:

1. **Hỗ trợ, không thay thế**: LLMs nên được dùng để giải thích khái niệm, đưa ví dụ hoặc phản hồi, chứ không phải để cung cấp luôn đáp án cuối cùng. Điều này giúp người học vẫn phải tự xử lý kiến thức.
2. **Khuyến khích tư duy phản biện**: Người học nên được hướng dẫn đặt câu hỏi, kiểm chứng, và mở rộng từ thông tin LLM đưa ra. Ví dụ: so sánh câu trả lời của LLM với sách giáo khoa hoặc tự đánh giá độ chính xác.
3. **Học tập thích ứng**: LLMs có thể cá nhân hóa trải nghiệm học, đưa ra lời giải thích phù hợp với trình độ từng người. Điều này hỗ trợ hiểu sâu hơn nhưng vẫn đòi hỏi sự tham gia chủ động.
4. **Minh bạch và định hướng**: Giáo viên cần giải thích rõ cho học sinh về điểm mạnh và hạn chế của LLMs, từ đó dạy cách sử dụng có trách nhiệm, tránh lệ thuộc.
5. **Học tập chủ động**: Sử dụng LLMs để tạo bài tập tương tác — như đóng vai hội thoại, giải bài tập, hoặc mô phỏng tình huống — để kích thích sự tham gia thay vì tiếp nhận thụ động.

Tóm lại, LLMs nên đóng vai trò **người hướng dẫn và cộng tác viên**, chứ không phải công cụ thay thế nỗ lực học tập của con người. Khi được tích hợp hợp lý, chúng có thể **tăng động lực, mở rộng khả năng tiếp cận tri thức và nuôi dưỡng sự tò mò**, đồng thời vẫn giữ được cốt lõi của việc học: **nỗ lực, suy ngẫm và tư duy**.