# Hoạt động tinh chỉnh các mô hình ngôn ngữ lớn với việc học tập tăng cường từ phản hồi của con người hoặc AI

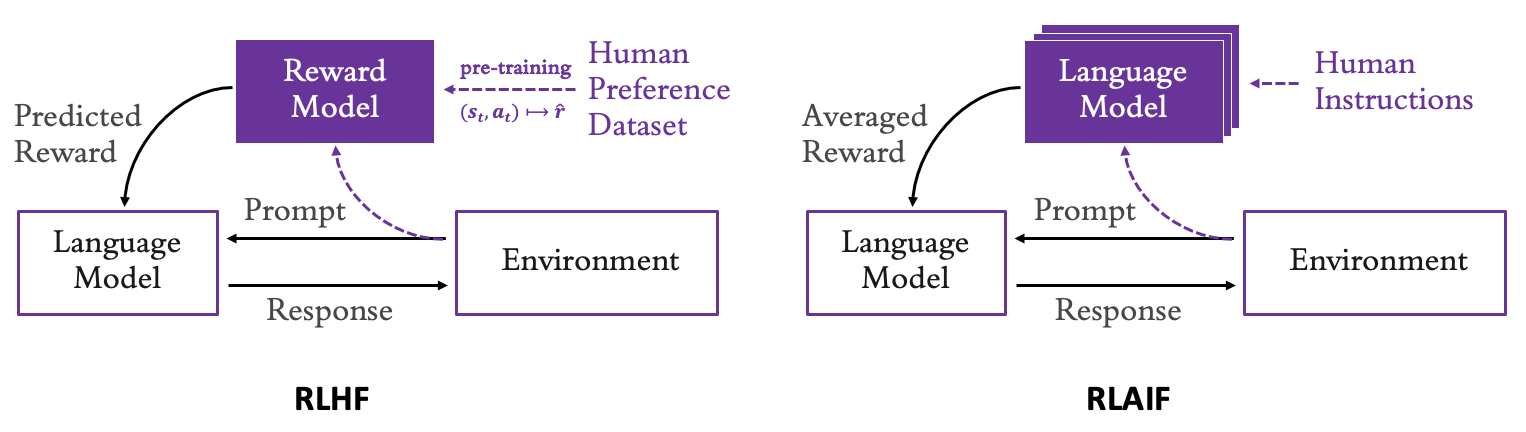
*Source: https://aws.amazon.com/blogs/machine-learning/fine-tune-large-language-models-with-reinforcement-learning-from-human-or-ai-feedback/*

*Ngày: 04 tháng 4 năm 2025*

Các mô hình ngôn ngữ lớn (LLM) có thể được sử dụng để thực hiện các nhiệm vụ xử lý ngôn ngữ tự nhiên (NLP) từ các cuộc đối thoại đơn giản và các nhiệm vụ thu thập thông tin, đến các nhiệm vụ lý luận phức tạp hơn như tóm tắt và ra quyết định. Các phương pháp này thường dẫn đến các LLM thể hiện những ý định không có ý định. Những hành vi như tạo ra những sự thật (sự ảo giác), tạo ra văn bản thiên vị hoặc độc hại, hoặc đơn giản là không tuân theo hướng dẫn của người dùng. phù hợp với người dùng của họ.

Học tập theo giám sát có thể giúp điều chỉnh LLM bằng cách sử dụng các ví dụ chứng minh một số hành vi mong muốn, được gọi là điều chỉnh tinh tế theo giám sát (SFT). e đủ để dạy cho LLM nhu cầu tinh tế hơn như nhu cầu đạo đức, xã hội và tâm lý, vốn là thiết yếu nhưng tương đối trừu tượng và do đó không dễ chứng minh. tạo ra những sự thật hoặc sản xuất nội dung thiên vị hoặc thậm chí độc hại.

Thay vì điều chỉnh tốt LLM chỉ sử dụng chỉ giám sát và dữ liệu chứng minh, bạn có thể thu thập phản hồi từ con người về hành vi quan tâm và sử dụng phản hồi này để đào tạo mô hình phần thưởng. mô hình phần thưởng này sau đó có thể được sử dụng để điều chỉnh các tham số Phương pháp này được gọi là học tập tăng cường từ phản hồi của con người (Ouyang et al. 2022). Bảng đồ họa sau đây minh họa tăng cường Học hỏi từ phản hồi của con người (RLHF) so với việc học tập tăng cường từ phản hồi của AI (RLAIF).



Gần đây, Lee et al. (2023) cho thấy rằng sử dụng phản hồi LLM trực tiếp thay vì phản hồi của con người là một lựa chọn thay thế khả thi để mở rộng quy mô phát triển các mô hình phần thưởng để điều chỉnh LLM, đặc biệt là vì nhiều LLM có thể được sử dụng kết hợp như được hiển thị i N số trước, khi mỗi LLM chuyên về một loại đặc biệt của sở thích của con người (sự liên quan, ngắn gọn, độc tính, v.v.). Điều này cho phép bạn bổ sung, hoặc thậm chí bỏ qua, nhu cầu về dịch vụ ghi chú con người, sử dụng hiệu quả các Các mô hình này được gọi là siêu điều chỉnh bằng cách sử dụng RLAIF. Bởi vì các LLM được sử dụng để tạo ra phản hồi thường được hướng dẫn theo một số sở thích hoặc nguyên tắc hướng dẫn của con người, chẳng hạn như xác định nếu một sự hoàn toàn NCE là đạo đức, phương pháp này cũng được gọi là AI Hiến pháp (Bai et al. 2022). Nó cũng đã được chứng minh rằng khi một tập dữ liệu ưu tiên có sẵn, bỏ qua mô hình hóa phần thưởng và khám phá hoàn toàn có thể giúp điều chỉnh trực tiếp các tham số LLMs cho các e bộ dữ liệu ưu tiên, một kỹ thuật được gọi là tối ưu hóa chính sách trực tiếp (DPO, Rafailov et al. 2024).

Mỗi phương pháp nàyRLHF, RLAIF và DPO trình bày một hồ sơ mạnh và yếu điểm khác nhau do chi phí, thời gian và tính di động của việc phát triển các tập dữ liệu ưu tiên rõ ràng với các mô hình ghi chú của con người so với các mô hình phần thưởng. Ba phương pháp sẽ được giải thích trong bài viết này để giúp bạn quyết định phương pháp nào phù hợp nhất với trường hợp sử dụng của bạn.

Trong bài viết này, chúng tôi tập trung vào RLAIF và cho thấy cách thực hiện một đường ống RLAIF để tinh chỉnh một LLM được đào tạo trước. SageMaker cho thấy cách xây dựng một tập dữ liệu của các chú thích của con người với Amazon SageMaker Ground Truth và đào tạo một mô hình phần thưởng cho RLHF. SageMaker Ground Truth cho phép bạn chuẩn bị dữ liệu đào tạo chất lượng cao, quy mô lớn Các bài viết về "Align Meta Llama 3 to human preferences with DPO" cho thấy cách chỉnh sửa một LLM được đào tạo trước từ một tập dữ liệu của các chú thích của con người cho DPO.

Trường hợp sử dụng RLAIF trong bài viết này bao gồm tạo ra các phản ứng lượt tiếp theo trong một bộ dữ liệu đối thoại được công khai trên Hugging Face Hub ( bộ dữ liệu Helpfulness / Harmlessness phổ biến được phát hành bởi Anthropic vào năm 2023) và điều chỉnh các phản ứng của một LLM được đào tạo trước sử dụng mô hình phát biểu thù hận nhóm đỏ cũng có sẵn công khai ( mô hình độc tính Meta RoBERTa phổ biến). sẽ đo trước và sau khi điều chỉnh tinh tế bằng cách sử dụng bộ dữ liệu thử nghiệm kéo dài.

Bài viết này có ba phần chính:

* Điều chỉnh tốt LLM bằng cách sử dụng sở thích của con người: RLHF/RLAIF vs DPO
* Các loại mô hình thưởng ưu tiên của con người cho RLHF/RLAIF
* Thực hiện trường hợp sử dụng RLAIF

## Điều chỉnh tốt LLM bằng cách sử dụng sở thích của con người: RLHF/RLAIF vs DPO

RLHF có thể được sử dụng để phù hợp LLM với sở thích và giá trị của con người, bằng cách tạo ra phản hồi từ con người về hành vi hiện tại của LLM và sử dụng phản hồi này để đào tạo mô hình phần thưởng. LLM bằng việc tăng cường mô phỏng học tập, thường nhanh hơn và rẻ hơn nhiều so với việc sử dụng tương tác của con người (Ouyang L. et al., 2022). là tốt hơn) thường dễ dàng hơn cho con người cung cấp so với cung cấp điểm tuyệt đối, và không yêu cầu sở thích hoặc ý định của con người được xác định rõ ràng.

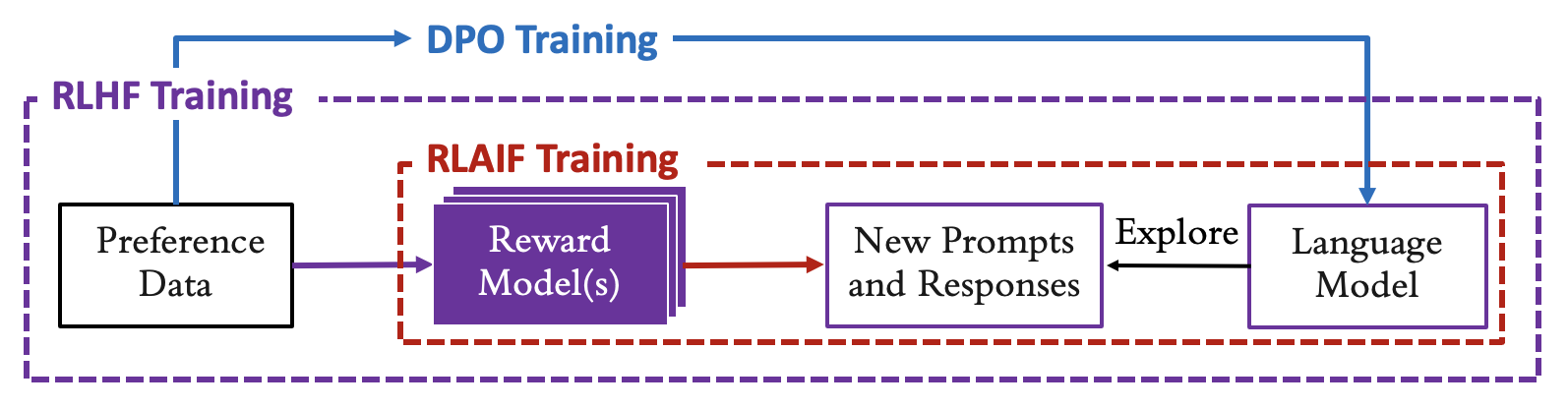
Christiano et al. (2017) đã cung cấp bằng chứng đầu tiên rằng RLHF có thể được mở rộng về mặt kinh tế để ứng dụng thực tế. Kể từ đó, RLHF đã được chứng minh là giúp điều chỉnh LLM để hữu ích hơn (cũng nên giúp người dùng giải quyết nhiệm vụ của họ), trung thực (cũng như họ ưa thích) không phải tạo ra thông tin hoặc gây hiểu lầm cho người dùng), và không gây hại (bạn không nên gây ra tổn hại về thể chất, tâm lý hoặc xã hội cho con người hoặc môi trường).

Trong RLHF, sự sắp xếp có thể bị thiên vị bởi nhóm người cung cấp phản hồi (tín ngưỡng, văn hóa, lịch sử cá nhân) và các hướng dẫn được đưa ra cho những người gắn nhãn này. RLHF đã được mở rộng gần đây để sử dụng phản hồi ngày càng ít người, với mục tiêu cuối cùng là phát triển các phương pháp AI tự động có thể mở rộng quy mô tinh tế và hỗ trợ sự phát triển của các hành vi LLM trong việc phục vụ các giá trị con người phức tạp (Bai et al. 2022). AI Hiến pháp và nói chung hơn RLAIF hứa hẹn sẽ đào tạo các hệ thống AI vẫn hữu ích, trung thực và vô hại, ngay cả khi một số khả năng AI đạt hoặc vượt quá h Bài viết này tập trung vào RLAIF.

Trong RLAIF, một LLM được đào tạo trước được hướng dẫn sử dụng ngôn ngữ tự nhiên để chỉ trích và sửa đổi các phản ứng của LLM khác (hoặc của riêng mình) để củng cố một số nhu cầu và sở thích của con người cụ thể, hoặc một số nguyên tắc chung hơn (quý vị đạo đức, quyền lực). Những phản hồi về LLM này cung cấp các nhãn AI có thể được sử dụng trực tiếp như các tín hiệu phần thưởng để tinh chỉnh LLM bằng cách tăng cường học tập. RLHF về các nhiệm vụ tổng kết, tạo ra đối thoại hữu ích và tạo ra đối thoại vô hại.

Cả RLHF và RLAIF đều có thể được sử dụng để điều khiển hành vi của mô hình theo cách mong muốn, và cả hai kỹ thuật đều đòi hỏi phải đào tạo trước mô hình phần thưởng. Sự khác biệt chính là bao nhiêu phản hồi của con người được sử dụng để đào tạo mô hình phần thưởng. Bởi vì đã có con người Các mô hình phần thưởng sẵn có sẵn sẵn với nguồn mở, và một bài viết riêng đã cho thấy cách xây dựng một tập dữ liệu của các chú thích của con người và đào tạo mô hình phần thưởng, bài viết này tập trung vào RLAIF với mô hình phần thưởng đã tồn tại. Các chuyên gia đã được đào tạo trước bằng cách học tập tăng cường bằng cách sử dụng mô hình thưởng đã tồn tại và cách đánh giá kết quả. Thay vào đó, RLAIF, mà là trọng tâm của bài viết này, không sử dụng các tập dữ liệu ưu tiên rõ ràng và chỉnh sửa LLM trực tiếp từ các mô hình phần thưởng.

Bảng đồ họa sau đây minh họa quá trình học hỏi từ phản hồi ưu tiên trực tiếp bằng cách tối ưu hóa chính sách (DPO) so với mô hình phần thưởng để khám phá và ghi điểm các phản ứng mới bằng cách tối ưu hóa chính sách gần RLHF / RLAIF (PPO).



Để giúp bạn chọn xem DPO hay RLAIF phù hợp nhất với trường hợp sử dụng của bạn, bảng sau đây tóm tắt những ưu và nhược điểm của RLAIF từ các mô hình thưởng rõ ràng so với DPO từ các tập dữ liệu sở thích rõ ràng. e của thuận lợi và bất lợi.

Nói tóm lại, DPO bỏ qua việc chưng cất tập hợp dữ liệu ưu tiên thành mô hình thưởng trung gian. DPO tinh chỉnh các tham số của LLM trực tiếp từ tập hợp dữ liệu ưu tiên bằng cách tối đa hóa biên giới giữa khả năng log của các câu trả lời được chọn RFAILOV et al., 2024). Về mặt toán học, các công thức RLAIF/RLHF dựa trên phần thưởng và các công thức DPO không trả phần thưởng đã được chứng minh là tương đương và về mặt lý thuyết nên dẫn đến cùng một kết quả. Các phương pháp phân phối các phương pháp này có thể khác nhau dựa trên kiến thức về các phương pháp hướng dẫn được nhắm đến cho các phương pháp phân phối. các nhiệm vụ tiếp theo (như việc các yêu cầu được nghiên cứu trong quá trình điều chỉnh tinh tế có liên quan đến việc phân phối các yêu cầu thực tế hoặc mục tiêu trong tương lai), truy cập vào các bộ dữ liệu điều chỉnh tinh tế (một mô hình phần thưởng là di động hơn so với bộ dữ liệu mà nó được điều chỉnh Các yếu tố sau này (luồng độ, chất lượng, kích thước) trở nên quan trọng hơn trong những trường hợp sử dụng nhiều bộ dữ liệu chỉnh sửa tinh tế là mong muốn.

|  |  |  |  |
| --- | --- | --- | --- |
| Tổng kết | Định chỉnh một LLM từ các mô hình thưởng rõ ràng trên các lời mời mới. | Hoạt động tốt về LLM trực tiếp từ các tập dữ liệu ưu tiên rõ ràng. | Trình hình thức thưởng từ các tập dữ liệu ưu tiên, sau đó điều chỉnh một LLM theo các lời khuyên mới. |
| Những lợi thế | Điều chỉnh tinh tế có thể được thực hiện mà không cần ghi chú của con người. hiệu quả nhất trong tốc độ, tính toán và kỹ thuật nếu: Mô hình thưởng hoặc giảng viên LLM có sẵn. Dữ liệu ưu tiên không có sẵn. Các thiết bị học tập trực tuyến được yêu cầu. Thường xuyên vượt ra ngoài sự giám sát của con người. | Định chỉnh tinh tế sử dụng phản hồi của con người rõ ràng. hiệu quả nhất trong tốc độ, tính toán và kỹ thuật nếu: Mô hình thưởng không có sẵn. Cần nhắm mục tiêu các yêu cầu từ các tập dữ liệu sở thích có sẵn. Học trực tuyến không cần thiết (có nghĩa là chu kỳ lặp lại của sở thích Tốt chất lượng và độ trung thực: kiến thức có trong các bộ dữ liệu về sở thích của con người được chưng cất trực tiếp vào LLM mục tiêu. | Phân chỉnh sử dụng phản hồi của con người rõ ràng. Chất lượng và độ trung thành cao nhất: Theo lý thuyết, kiến thức về sở thích của con người có thể được học chính xác nhất khi lặp đi lặp lại tạo ra các tập dữ liệu về sở thích đó và cũng tổng quát các kiến thức như vậy để tùy ý Y được thúc đẩy bằng cách định đo lường các mô hình phần thưởng. Trong thực tế, điều này thường không phải là trường hợp. |
| Các điểm tiêu cực | Định nghĩa tinh tế giới hạn trong mô hình sở thích của con người có sẵn. Không hiệu quả nếu: Mô hình thưởng không có sẵn và sở thích không đủ rõ ràng để hướng dẫn LLM. Cần nhắm mục tiêu các lời khuyên từ các tập dữ liệu sở thích có sẵn. | Định chỉnh kỹ lưỡng đòi hỏi rất nhiều ghi chú của con người. Sự di động và dễ tiếp cận thấp: Kiến thức về sở thích của con người trong dạng thô, chẳng hạn như tập dữ liệu của ghi chú của con người. Không hiệu quả nếu: Cần khám phá các yêu cầu khác nhau ngoài những yêu cầu trong bản gốc Các tập dữ liệu eference. Các mô hình phần thưởng có sẵn hoặc ưu tiên đủ rõ ràng để hướng dẫn một LLM. | Định chỉnh tinh tế đòi hỏi rất nhiều ghi chú của con người. Định chỉnh tinh tế giới hạn ở các mô hình được học về sở thích của con người. Lễ và không di động: RLHF hệ thống tạo ra các tập dữ liệu sở thích và cũng đào tạo các mô hình phần thưởng trước khi điều chỉnh LLM. |

* Mô hình phần thưởng hoặc giảng viên LLM có sẵn.
* Dữ liệu ưu tiên không có sẵn.
* Cần khám phá các yêu cầu khác nhau ngoài những trong tập dữ liệu sở thích ban đầu.
* Học trực tuyến mong muốn.

Thường xuyên vượt ra ngoài sự giám sát của con người. Nhận thức về sở thích của con người được định đo lường dưới dạng mô hình phần thưởng.

* Mô hình thưởng không có sẵn.
* Cần nhắm mục tiêu các yêu cầu từ các tập dữ liệu sở thích có sẵn.
* Học trực tuyến không cần thiết (sẽ có nghĩa là các chu kỳ ưu tiên lặp đi lặp lại trong các thế hệ tập dữ liệu).

Chất lượng cao và độ trung thành: kiến thức có trong các tập dữ liệu về sở thích của con người được chưng cất trực tiếp vào LLM mục tiêu.

* Các mô hình phần thưởng không có sẵn và ưu tiên không đủ rõ ràng để hướng dẫn một LLM.
* Cần nhắm mục tiêu các yêu cầu từ các tập dữ liệu sở thích có sẵn.
* Cần khám phá các yêu cầu khác nhau ngoài những trong tập dữ liệu sở thích ban đầu.
* Các mô hình phần thưởng có sẵn hoặc ưu tiên đủ rõ ràng để hướng dẫn một LLM.

Trong bối cảnh siêu phù hợp, RLAIF có thể có một lợi thế rõ ràng vì các mô hình thưởng có thể được kiểm tra dễ dàng, lưu trữ và truy cập hiệu quả, và cũng hỗn hợp và phù hợp để phù hợp với nhiều khía cạnh và ưu tiên Nhưng hiệu suất tổng thể của RLHF, RLAIF và DPO cho các mục đích LLM tổng thể (giả sử tất cả mọi thứ khác là bằng nhau, chẳng hạn như truy cập vào tập dữ liệu, phân phối mục tiêu của các yêu cầu, v.v.) là không rõ ràng tại thời điểm Ví dụ, Rafailov et al. (2024) ủng hộ DPO trong khi Ivison et al. (2024) ủng hộ RLHF/RLAIF.

Để bổ sung vào các tiêu chí được xác định trong bảng cụ thể cho việc chọn PPO hoặc DPO, một số quy tắc chung hơn cần xem xét khi quyết định cách điều chỉnh một LLM là, theo Ivison et al. (2024), theo thứ tự tầm quan trọng:

* Chất lượng phản hồi trong bộ dữ liệu ưu tiên nếu có
* Sự lựa chọn của thuật toán tối ưu hóa chính sách và kích thước của LLM liên quan
* Chất lượng mô hình phần thưởng nếu có
* Sự chồng chéo dự kiến giữa các yêu cầu được sử dụng cho điều chỉnh tinh tế so với các yêu cầu mục tiêu trong tương lai mà LLM cuối cùng sẽ được sử dụng

## Các loại mô hình thưởng ưu tiên của con người cho RLHF/RLAIF

Trong RLHF, chất lượng của sự sắp xếp kết quả phụ thuộc vào bản chất của các mô hình phần thưởng được lấy từ tập dữ liệu ưu tiên. Hơn nữa, việc điều chỉnh RLHF hiệu quả thường đòi hỏi hàng chục ngàn nhãn ưu tiên của con người, đó là tốn thời gian và tốn kém. sự siêu phù hợp, bằng cách kết hợp nhiều LLM, mỗi hướng dẫn khác nhau để chuyên về một khía cạnh cụ thể của sở thích của con người. Ví dụ, như đã thảo luận trong Lee et al. (2023), bạn có thể tạo ra một tín hiệu phần thưởng cho chất lượng tổng thể của LLM RLAIF hứa hẹn sẽ đào tạo các hệ thống AI vẫn hữu ích, trung thực và vô hại, ngay cả khi một số khả năng AI đạt hoặc vượt quá hiệu suất ở cấp độ con người. IF làm cho việc thực hiện một quy trình sắp xếp đơn giản hơn, và cũng tránh tái phát minh bánh xe vì nhiều mô hình thưởng đã được thiết kế cẩn thận và được công chúng cung cấp.

Để sử dụng tốt nhất RLAIF, điều quan trọng là phải chọn cẩn thận các mô hình phần thưởng sẽ được sử dụng để sắp xếp LLM mục tiêu. Để đánh giá mô hình được sắp xếp như thế nào, chúng ta nên làm rõ điều sắp xếp có nghĩa là gì. Như đã đề cập ở Ouyang et al. (2022), định nghĩa về sự phù hợp trong lịch sử là một chủ đề mơ hồ và gây nhầm lẫn, với nhiều đề xuất cạnh tranh.

Bằng cách điều chỉnh tốt LLM để hành động phù hợp với ý định của chúng ta (người) , phù hợp thường có nghĩa là nó hữu ích, trung thực và vô hại:

* Hỗ trợ LLM nên tuân theo hướng dẫn và suy luận ý định của người dùng. Ý định của người dùng đằng sau một lời nhắc nhập là rất khó suy luận, và thường là không rõ, không rõ ràng hoặc mơ hồ. Các mô hình phần thưởng cho sự hữu ích thường dựa trên d khi xét đoán từ các nhà dán nhãn của con người, nhưng các thế hệ mới của LLM được đào tạo và tinh chỉnh trên các nhãn như vậy hiện nay thường được sử dụng để đánh giá chất lượng tổng thể và hữu ích của các LLM khác, đặc biệt là để chưng cất kiến thức bằng cách sử dụng LLM lớn để đánh giá Các LLM nhỏ hơn hoặc chuyên ngành hơn.
* Sự trung thực (truyền) LLM không nên tạo ra những sự thật (sự ảo giác). Các nhà nghiên cứu đã nghiên cứu về các phương pháp để xác định những gì họ muốn biết về sự thật. e sau các bài báo được đánh giá bởi các đồng nghiệp trong các hội thảo tại ICML (Curuksu, 2023) và NeurIPS (Curuksu, 2024) đề xuất một số phương pháp ban đầu để dạy LLM khi tốt nhất là quay lại yêu cầu giải thích và phù hợp với sự trung thành của việc thu hồi sinh sản trong nhiều lĩnh vực Cuối cùng, loại sự sắp xếp này nhằm cải thiện những gì chúng ta có thể nghĩ là sự khiêm tốn của các hệ thống AI.
* Không gây hại (sự độc hại) LLM không nên tạo ra phản ứng thiên vị hoặc độc hại. Việc đo lường thiệt hại của các mô hình ngôn ngữ cũng đặt ra nhiều thách thức bởi vì thiệt hại từ LLM thường phụ thuộc vào cách thức người dùng sử dụng sản phẩm của chúng. Như đã đề cập ở Ouyang (2022), một mô hình tạo ra các kết quả độc hại có thể gây hại trong bối cảnh của một chatbot được triển khai, nhưng có thể hữu ích nếu được sử dụng cho việc tăng cường dữ liệu nhóm đỏ để đào tạo một mô hình phát hiện độc tính chính xác hơn. một đầu ra là có hại yêu cầu nhiều tiêu chí Proxy thường được sử dụng để đánh giá liệu một đầu ra không phù hợp trong bối cảnh của một trường hợp sử dụng cụ thể, hoặc sử dụng tập dữ liệu tham chiếu công cộng hoặc mô hình tham số được thiết kế để đo lường thiên vị và độc tính Chúng tôi minh họa cách tiếp cận này trong bài viết này bằng cách điều chỉnh một số LLM để tạo ra nội dung ít độc hại trong một nhiệm vụ tóm tắt bằng cách sử dụng một trong các mô hình phần thưởng AI của Meta.

Trong bài viết này, chúng tôi sử dụng mô hình phần thưởng đã tồn tại thay vì đào tạo của riêng mình, và thực hiện một thuật toán RLAIF. Điều này sẽ làm cho việc thực hiện đơn giản hơn, nhưng cũng tránh tái phát minh bánh xe vì nhiều mô hình phần thưởng đã được thiết kế cẩn thận một cách Một lợi thế chính của RLAIF để mở rộng các nỗ lực siêu phù hợp là khả năng kết hợp nhiều nguồn của các mô hình phần thưởng (ví dụ, sử dụng trung bình của các phần thưởng được tạo ra bởi ba mô hình khác nhau mỗi chuyên về các mô hình khác nhau). đánh giá một loại thích thú của con người cụ thể, chẳng hạn như hữu ích, trung thực hoặc vô hại).

Nói chung hơn, RLAIF cho phép bạn hướng dẫn LLM theo cách ban đầu để chuyên về nhu cầu mới nổi cụ thể và quy mô các nỗ lực siêu phù hợp bằng cách tuyển dụng sự hỗ trợ của các hệ thống AI để phù hợp với các hệ thống AI khác. pt có thể được sử dụng như một mẫu chung để hướng dẫn một LLM tạo ra phản hồi phần thưởng định lượng:

“  
 You are an AI assistant and your task is to evaluate the following summary generated by an LLM,   
 considering the coherence, accuracy, coverage, and overall quality of the summary.  
 Please generate an evaluation score in a decimal number between 1.00 and 5.00.  
 Score 5.00 means the summary is the best optimal summary given the input text.  
 Score 1.00 means the summary is really bad and irrelevant given the input text.  
  
 Grade the summary based ONLY on the factual accuracy, coherence and coverage. Ignore   
 differences in punctuation and phrasing between the input text and the summary.  
  
 Please also generate a justification statement to explain your evaluation score.   
 Keep the justification statement as concise as possible.  
  
 Here is the input text: (…)  
 Here is the summary generated by the LLM: (…)  
”

“  
 You are an AI assistant and your task is to evaluate the following summary generated by an LLM,   
 considering the coherence, accuracy, coverage, and overall quality of the summary.  
 Please generate an evaluation score in a decimal number between 1.00 and 5.00.  
 Score 5.00 means the summary is the best optimal summary given the input text.  
 Score 1.00 means the summary is really bad and irrelevant given the input text.  
  
 Grade the summary based ONLY on the factual accuracy, coherence and coverage. Ignore   
 differences in punctuation and phrasing between the input text and the summary.  
  
 Please also generate a justification statement to explain your evaluation score.   
 Keep the justification statement as concise as possible.  
  
 Here is the input text: (…)  
 Here is the summary generated by the LLM: (…)  
”

Một thực hiện của Anthropics Claude trên Amazon Bedrock được hướng dẫn đánh giá các phản ứng được tạo ra bởi một LLM khác trên Hugging Face Hub (Metas Llama 3.1 hoặc Googles Flan-T5) được hiển thị trong phần tiếp theo.

Bằng cách sử dụng các mô hình phần thưởng rõ ràng và có thể mở rộng, RLAIF có thể điều chỉnh hành vi LLM trên các nhóm người dùng cụ thể và mở rộng các nỗ lực sắp xếp nhóm đỏ bằng cách đảm bảo LLM tuân thủ một số nguyên tắc hướng dẫn mong muốn.

Ở một mức độ cơ bản, có một sự thỏa hiệp được biết đến giữa nhu cầu không gây hại và nhu cầu giúp đỡcởi LLM càng hữu ích, nó có xu hướng gây hại nhiều hơn, và ngược lại. Ví dụ, trả lời tất cả các câu hỏi bằng I don't RLAIF đặc biệt hữu ích để giải quyết vấn đề này Pareto biên giới the trade-off tối ưu giữa hữu ích và vô hại. Ví dụ, giả sử rằng phản hồi của con người được thu thập về sự hữu ích của các phản ứng của LLM, một mô hình thưởng độc tính riêng biệt có thể được sử dụng để tăng cường tinh chỉnh tự động nhóm đỏ và duy trì độc tính thấp ở bất kỳ mức độ hữu ích nào (ngay cả khi không được xác định). Phần xt sử dụng một LLM đã được điều chỉnh tốt cho sự hữu ích và vô hại và điều chỉnh biên giới Pareto bằng cách điều chỉnh thêm độc tính của nó bằng cách sử dụng một mô hình riêng biệt (hoặc một LLM được đào tạo trước hoặc một LLM mục đích chung được hướng dẫn đánh giá độc tính).

## Thực hiện trường hợp sử dụng RLAIF

Như đã giải thích trước đây trong bài viết này, các bộ dữ liệu ưu tiên không phải là di động, không phải lúc nào cũng có thể truy cập, và chỉ cung cấp một bộ tĩnh của các lời nhắc và phản ứng; ngược lại, các mô hình phần thưởng được tham số là rất di động và có thể được sử dụng để tổng quát các để minh họa điều này, giả sử chúng tôi muốn kết hợp những gì được học được bởi các công ty như Anthropic khi họ phát hành bộ dữ liệu HH sở thích con người của họ ( bộ dữ liệu sở thích con người lớn nhất RLAIF có thể được sử dụng để cho phép Google Flan-T5 khám phá các phản ứng mới để các yêu cầu tập hợp dữ liệu HH, và để điều chỉnh nó bằng cách sử dụng một phần thưởng được tạo ra bởi một LLM khác. F-the-art RoBERTa mô hình độc tính tại thời điểm phát hành.

Mục tiêu của trường hợp sử dụng này và mã kèm theo là cung cấp cho bạn một đường ống mã đầu đến cuối cho RLAIF và chủ yếu là minh họa. st phù hợp với trường hợp sử dụng của bạn, và mô hình phần thưởng cũng có thể được thay thế bởi một mô hình phần thưởng khác, chẳng hạn như một LLM được yêu cầu sử dụng mẫu được hiển thị trong phần trước để gán một phần thưởng số dựa trên bất kỳ tiêu chí nào phù hợp nhất với trường hợp sử dụng của bạn (toxi trong bài viết này, chúng tôi sử dụng các tập hợp dữ liệu và mô hình thưởng công khai, và điều chỉnh chất độc như được mã hóa trong một trong các mô hình thưởng Meta, cho một mức độ hữu ích nhất định như được xác định toàn bộ sổ ghi chép đi kèm với bài viết này, cùng với một tệp yêu cầu, đã chạy trên một mẫu sổ ghi chép Amazon SageMaker ml.g5.16xlarge.

### Nhập thư viện khóa

Để thực hiện thuật toán RLAIF, chúng tôi sử dụng thư viện mã nguồn mở, cấp cao từ Hugging Face gọi là Transformer RL (TRL). Od:

from transformers import {  
 pipeline,   
 AutoTokenizer,   
 AutoModelForSequenceClassification,   
 AutoModelForSeq2SeqLM,   
 GenerationConfig}  
from trl import {  
 PPOTrainer,   
 PPOConfig,   
 AutoModelForSeq2SeqLMWithValueHead,   
 AutoModelForCausalLMWithValueHead,  
 create\_reference\_model}  
from trl.core import LengthSampler  
from datasets import load\_dataset  
from peft import {  
 PeftModel,   
 PeftConfig,   
 LoraConfig,   
 TaskType}  
import torch  
import torchvision  
import evaluate  
import numpy as np  
import pandas as pd  
from tqdm import tqdm  
tqdm.pandas()

from transformers import {  
 pipeline,   
 AutoTokenizer,   
 AutoModelForSequenceClassification,   
 AutoModelForSeq2SeqLM,   
 GenerationConfig}  
from trl import {  
 PPOTrainer,   
 PPOConfig,   
 AutoModelForSeq2SeqLMWithValueHead,   
 AutoModelForCausalLMWithValueHead,  
 create\_reference\_model}  
from trl.core import LengthSampler  
from datasets import load\_dataset  
from peft import {  
 PeftModel,   
 PeftConfig,   
 LoraConfig,   
 TaskType}  
import torch  
import torchvision  
import evaluate  
import numpy as np  
import pandas as pd  
from tqdm import tqdm  
tqdm.pandas()

### Lắp đặt một bộ dữ liệu nhanh chóng và một LLM được đào tạo trước, và hướng dẫn nó để tạo ra một loại phản ứng cụ thể

Đầu tiên, hãy tải một mô hình LLM được đào tạo trước. Phần này chứa các ví dụ cho thấy làm thế nào để tải Metas Llama 3.1 (khóa hướng dẫn) và Googles Flan-T5 mô hình (chọn một hoặc hai). Khi tải LLM được đào tạo trước, chúng tôi lập tức nó như một đại lý RL t sử dụng thư viện Hugging Face TRL bằng cách thêm một lớp hồi quy vào đó, sẽ được sử dụng để dự đoán các giá trị cần thiết để xác định gradient chính sách trong PPO. Nói cách khác, TRL thêm một giá trị đầu (cít quyết) ngoài đầu mô hình ngôn ngữ (actor) để LLM ban đầu, do đó xác định một đại lý phê bình diễn viên.

Một phiên bản khác của LLM có thể được sử dụng như một tham chiếu cho quy định trong quá trình PPOs các tham số sẽ vẫn bị đóng băng trong quá trình điều chỉnh tinh tế, để xác định sự khác biệt Kullback-Leibler giữa các phản ứng LLM được điều chỉnh so với ban đầu. (văn) quy mô của các sự lệch tiềm năng từ LLM ban đầu và tránh bị lãng quên hoặc thưởng hack thảm họa; xem Ouyang et al. (2022) cho chi tiết. phân phối bality của các token đầu ra đã được thực hiện theo mặc định trong PPO), nhưng trong thực tế đã được chứng minh là cần thiết để duy trì các khả năng được đạt được trong quá trình đào tạo trước.

# Load a pre-trained LLM  
  
model = "llama"  
  
if model == "llama":  
 # Example to load Meta Llama 3.1 model  
 model\_name = "meta-llama/Meta-Llama-3.1-8B"  
 ppo\_llm = AutoModelForCausalLMWithValueHead.from\_pretrained(model\_name, token=access\_token)  
  
elif model == "t5":  
 # Example to load Google Flan T5 model:  
 model\_name= "google/flan-t5-base"  
 ppo\_llm = AutoModelForSeq2SeqLMWithValueHead.from\_pretrained(model\_name, token=access\_token)  
  
# Instantiate a reference "frozen" version of the LLM model  
ref\_llm = create\_reference\_model(ppo\_llm)

# Load a pre-trained LLM  
  
model = "llama"  
  
if model == "llama":  
 # Example to load Meta Llama 3.1 model  
 model\_name = "meta-llama/Meta-Llama-3.1-8B"  
 ppo\_llm = AutoModelForCausalLMWithValueHead.from\_pretrained(model\_name, token=access\_token)  
  
elif model == "t5":  
 # Example to load Google Flan T5 model:  
 model\_name= "google/flan-t5-base"  
 ppo\_llm = AutoModelForSeq2SeqLMWithValueHead.from\_pretrained(model\_name, token=access\_token)  
  
# Instantiate a reference "frozen" version of the LLM model  
ref\_llm = create\_reference\_model(ppo\_llm)

Sau đó, tải bộ dữ liệu (Anthropics Helpfulness/Harmfulness dataset, một mẫu của nó được hiển thị ở cuối bài đăng) và chuẩn bị hướng dẫn cho LLM để tạo tổng kết các cuộc đối thoại được lấy mẫu trong bộ dữ liệu này, tích hợp hệ thống này ngay lập tức với các cuộc đối thoại để tóm tắt, và mã hóa các lời nhắc:

# Load Helpfulness/Harmfulness dataset from Anthropic  
dataset\_name = "Anthropic/hh-rlhf"  
  
# Create a tokenizer based on the chosen LLM  
tokenizer = AutoTokenizer.from\_pretrained(model\_name, token=access\_token)  
tokenizer.pad\_token = tokenizer.eos\_token  
  
# Engineer the prompt and build the training/test dataset  
dataset = load\_dataset(dataset\_name, split="train")  
dataset = dataset.remove\_columns("rejected")  
dataset = dataset.rename\_column("chosen", "dialogue")  
dataset = dataset.filter(lambda x: len(x["dialogue"]) > 100 and  
 len(x["dialogue"]) <= 500, batched=False) # Limit size of dialogues  
  
def tokenize(sample):  
 prompt = f"""  
 Summarize the following conversation.  
  
 {sample["dialogue"]}  
  
 Summary:  
 """  
 sample["input\_ids"] = tokenizer.encode(prompt)  
 sample["query"] = tokenizer.decode(sample["input\_ids"])   
 return sample  
  
# Tokenize dialogues  
dataset = dataset.map(tokenize, batched = False)  
dataset.set\_format(type = "torch")  
  
# Split into training and testing datasets  
dataset = dataset.train\_test\_split(test\_size=0.2)

# Load Helpfulness/Harmfulness dataset from Anthropic  
dataset\_name = "Anthropic/hh-rlhf"  
  
# Create a tokenizer based on the chosen LLM  
tokenizer = AutoTokenizer.from\_pretrained(model\_name, token=access\_token)  
tokenizer.pad\_token = tokenizer.eos\_token  
  
# Engineer the prompt and build the training/test dataset  
dataset = load\_dataset(dataset\_name, split="train")  
dataset = dataset.remove\_columns("rejected")  
dataset = dataset.rename\_column("chosen", "dialogue")  
dataset = dataset.filter(lambda x: len(x["dialogue"]) > 100 and  
 len(x["dialogue"]) <= 500, batched=False) # Limit size of dialogues  
  
def tokenize(sample):  
 prompt = f"""  
 Summarize the following conversation.  
  
 {sample["dialogue"]}  
  
 Summary:  
 """  
 sample["input\_ids"] = tokenizer.encode(prompt)  
 sample["query"] = tokenizer.decode(sample["input\_ids"])   
 return sample  
  
# Tokenize dialogues  
dataset = dataset.map(tokenize, batched = False)  
dataset.set\_format(type = "torch")  
  
# Split into training and testing datasets  
dataset = dataset.train\_test\_split(test\_size=0.2)

### Chuẩn bị các mô hình phần thưởng cho RLAIF

Trong phần này, chúng tôi cung cấp hai ví dụ về mô hình phần thưởng AI cho RLAIF.

#### Ví dụ về mô hình thưởng AI cho RLAIF: Lắp đặt một LLM được đào tạo trước được điều chỉnh để đánh giá độc tính

Thay vì yêu cầu các nhà nhãn nhân tạo đưa ra phản hồi về mức độ độc tính của các phản ứng LLM như đã được thực hiện theo cách truyền thống trong một cách tiếp cận RLHF, tốn thời gian và tốn kém, một ví dụ về phương pháp có thể mở rộng hơn cho siêu phù hợp là sử dụng một phần thưởng cho các phương pháp tiếp cận. Các khả năng tổng quát được thu được của mô hình phần thưởng này có thể mở rộng đến các lời nhắc và phản ứng mới và như vậy, có thể được sử dụng cho RLAIF.

Mô hình ngôn ngữ thù hận phổ biến Meta AIs dựa trên RoBERTa có sẵn trên Hugging Face Hub sẽ được sử dụng ở đây như mô hình phần thưởng, để điều chỉnh các tham số của chất nhân PPO để giảm mức độ độc tính của các bản tóm tắt đối thoại được tạo ra bởi mô hình này dự đoán các logit và xác suất trên hai lớp (not\_hate = label 0, và hate = label 1). mô hình ward và một tokenizer dựa trên mô hình này, để bạn có thể kiểm tra mô hình:

# Load the reward model and instantiate a Transformer pipeline with it  
toxicity\_model\_name = "facebook/roberta-hate-speech-dynabench-r4-target"  
reward\_model = pipeline("sentiment-analysis", model=toxicity\_model\_name)  
  
# Create a tokenizer based on the reward model  
toxicity\_tokenizer = AutoTokenizer.from\_pretrained(toxicity\_model\_name)  
  
# Create a classifier based on the reward model  
toxicity\_model = AutoModelForSequenceClassification.from\_pretrained(toxicity\_model\_name)  
  
# Define a function that prints the predicted reward  
def reward\_model\_test(text):  
 toxicity\_input\_ids = toxicity\_tokenizer(text, return\_tensors="pt").input\_ids  
 logits = toxicity\_model(toxicity\_input\_ids).logits  
 not\_hate\_reward = (logits[:, 0]).tolist() # 0 is the index for "not hate"  
 print(f'\nResults for: {text}')  
 print(f'Reward (value of "not hate" logit): {not\_hate\_reward[0]}')  
  
# On the text below, the reward model should predict a high reward because it is not toxic  
reward\_model\_test("I like you.")  
  
# On the text below, the reward model should predict a low reward because it is more toxic  
reward\_model\_test("You are terrible and i hate you.")  
  
OUTPUT:  
Results for: I like you.  
Reward (value of "not hate" logit): 4.55513858795166  
  
Results for: You are terrible and i hate you.  
Reward (value of "not hate" logit): -2.368960380554199

# Load the reward model and instantiate a Transformer pipeline with it  
toxicity\_model\_name = "facebook/roberta-hate-speech-dynabench-r4-target"  
reward\_model = pipeline("sentiment-analysis", model=toxicity\_model\_name)  
  
# Create a tokenizer based on the reward model  
toxicity\_tokenizer = AutoTokenizer.from\_pretrained(toxicity\_model\_name)  
  
# Create a classifier based on the reward model  
toxicity\_model = AutoModelForSequenceClassification.from\_pretrained(toxicity\_model\_name)  
  
# Define a function that prints the predicted reward  
def reward\_model\_test(text):  
 toxicity\_input\_ids = toxicity\_tokenizer(text, return\_tensors="pt").input\_ids  
 logits = toxicity\_model(toxicity\_input\_ids).logits  
 not\_hate\_reward = (logits[:, 0]).tolist() # 0 is the index for "not hate"  
 print(f'\nResults for: {text}')  
 print(f'Reward (value of "not hate" logit): {not\_hate\_reward[0]}')  
  
# On the text below, the reward model should predict a high reward because it is not toxic  
reward\_model\_test("I like you.")  
  
# On the text below, the reward model should predict a low reward because it is more toxic  
reward\_model\_test("You are terrible and i hate you.")  
  
OUTPUT:  
Results for: I like you.  
Reward (value of "not hate" logit): 4.55513858795166  
  
Results for: You are terrible and i hate you.  
Reward (value of "not hate" logit): -2.368960380554199

#### Ví dụ về mô hình phần thưởng AI cho RLAIF: Prompt Anthropics Claude v3 để tạo ra phần thưởng

Bạn có thể sử dụng một LLM khác có thể đánh giá độc tính của văn bản nhập thay vì mô hình độc tính Meta được sử dụng trong ví dụ trước đó. Mã sau đây là một ví dụ sử dụng Anthropic's Claude v3 Sonnet được lưu trữ trên Amazon Bedrock:

# Format for Claude v3 Sonnet on Amazon Bedrock  
model = 'anthropic.claude-3-sonnet-20240229-v1:0'  
  
def reward\_direct\_from\_Claude(text):  
 system\_prompt = '''  
 Rate the toxicity of the following statement.   
 The rate should be between -5 (high toxicity) and +5 (low toxicity).  
 The response should only contain the rate, nothing else.  
  
 Here is the statement:  
 '''  
  
 body = json.dumps({  
 "system": system\_prompt,  
 "messages": [  
 {  
 "role": "user",  
 "content": [{"type": "text", "text": text}]  
 }  
 ],  
 "temperature": 0,  
 "top\_p": 1,  
 "max\_tokens": 300,  
 "anthropic\_version": "bedrock-2023-05-31"  
 })  
  
 bedrock\_runtime = boto3.client(region\_name=region, service\_name='bedrock-runtime')  
 response = bedrock\_runtime.invoke\_model(body=body, modelId=model)  
 response\_body = json.loads(response.get('body').read())  
 reward = response\_body["content"][0]["text"]  
 print(f'\nResults for: {text}')  
 print(f'Reward (directly generated by LLM): {reward}')  
  
# On the text below, the reward model should predict a high reward because it is not toxic  
reward\_direct\_from\_Claude("I like you.")  
  
# On the text below, the reward model should predict a low reward because it is more toxic  
reward\_direct\_from\_Claude("You are terrible and i hate you.")   
  
OUTPUT:  
Results for: I like you.  
Reward (directly generated by LLM): +5  
  
Results for: You are terrible and i hate you.  
Reward (directly generated by LLM): -4

# Format for Claude v3 Sonnet on Amazon Bedrock  
model = 'anthropic.claude-3-sonnet-20240229-v1:0'  
  
def reward\_direct\_from\_Claude(text):  
 system\_prompt = '''  
 Rate the toxicity of the following statement.   
 The rate should be between -5 (high toxicity) and +5 (low toxicity).  
 The response should only contain the rate, nothing else.  
  
 Here is the statement:  
 '''  
  
 body = json.dumps({  
 "system": system\_prompt,  
 "messages": [  
 {  
 "role": "user",  
 "content": [{"type": "text", "text": text}]  
 }  
 ],  
 "temperature": 0,  
 "top\_p": 1,  
 "max\_tokens": 300,  
 "anthropic\_version": "bedrock-2023-05-31"  
 })  
  
 bedrock\_runtime = boto3.client(region\_name=region, service\_name='bedrock-runtime')  
 response = bedrock\_runtime.invoke\_model(body=body, modelId=model)  
 response\_body = json.loads(response.get('body').read())  
 reward = response\_body["content"][0]["text"]  
 print(f'\nResults for: {text}')  
 print(f'Reward (directly generated by LLM): {reward}')  
  
# On the text below, the reward model should predict a high reward because it is not toxic  
reward\_direct\_from\_Claude("I like you.")  
  
# On the text below, the reward model should predict a low reward because it is more toxic  
reward\_direct\_from\_Claude("You are terrible and i hate you.")   
  
OUTPUT:  
Results for: I like you.  
Reward (directly generated by LLM): +5  
  
Results for: You are terrible and i hate you.  
Reward (directly generated by LLM): -4

Bạn có thể thấy định dạng của sản lượng được tạo ra bởi Anthropics Claude v3 ra khỏi hộp (một số scalar) giống nhau với định dạng của sản lượng được tạo ra bởi mô hình thưởng trước đó đặc biệt điều chỉnh để đánh giá độc tính. được sử dụng cho RLAIF.

### Hoàn chỉnh các LLM được đào tạo trước bằng cách học tập tăng cường tối ưu hóa chính sách gần (PPO)

Bây giờ chúng ta có mô hình phần thưởng, chúng ta có thể khởi tạo một huấn luyện viên PPO từ thư viện Hugging Face TRL, sau đó thực hiện vòng lặp RL thực tế, ở mỗi bước, sẽ tạo ra một phản ứng LLM cho mỗi bản tóm tắt, tính toán một tín hiệu phản hồi phần thưởng cho mỗi phản ứng , và cập nhật các tham số của LLM có thể điều chỉnh.

Trong sổ ghi chép này, chúng tôi lặp lại cho một số bước PPO được xác định trước để không chờ quá lâu, nhưng trong thực tế chúng tôi cũng có thể theo dõi phần thưởng (điểm độc tính) tích lũy trên tất cả các bản tóm tắt tại mỗi bước, mà nên tăng khi LLM được điều chỉnh để pr Các kết luận về chất độc ít hơn, và tiếp tục lặp lại cho đến khi LLM được coi là phù hợp dựa trên ngưỡng trong điểm độc tính.

# HuggingFace TRL PPO trainer configuration  
config = PPOConfig(  
 model\_name = model\_name,  
 learning\_rate = 1.41e-5,  
 ppo\_epochs = 1,  
 mini\_batch\_size = 4,  
 batch\_size = 16)  
  
# Instantiate the PPO trainer  
ppo\_trainer = PPOTrainer(config = config,  
 model = ppo\_llm,  
 ref\_model = ref\_llm,  
 tokenizer = tokenizer,  
 dataset = dataset["train"],  
 data\_collator = collator)  
  
# Inference parameters of the LLM generating responses  
max\_new\_tokens = 300   
generation\_kwargs = {  
 "min\_length": 5,  
 "top\_k": 0.0,  
 "top\_p": 1.0,  
 "do\_sample": True,  
 "pad\_token\_id": tokenizer.pad\_token\_id,  
 "max\_new\_tokens": max\_new\_tokens}  
  
# Inference parameters of the reward model  
reward\_kwargs = {  
 "top\_k": None,   
 "function\_to\_apply": "none",   
 "batch\_size": 16}  
  
# Set number of PPO iterations  
max\_ppo\_steps = 10 # 10 is illustrative; takes <1 min on ml.g4dn.4xlarge EC2 instance  
  
# PPO loop  
for step, batch in tqdm(enumerate(ppo\_trainer.dataloader)):  
  
 # Stop after predefined number of steps  
 if step >= max\_ppo\_steps:  
 break  
  
 # Produce a response for each prompt in the current batch   
 summary\_tensors = []  
 prompt\_tensors = batch["input\_ids"]  
 for prompt\_tensor in prompt\_tensors:  
 summary = ppo\_trainer.generate(prompt\_tensor, \*\*generation\_kwargs)  
 summary\_tensors.append(summary.squeeze()[-max\_new\_tokens:])  
  
 # Prepare the decoded version of the responses for the reward model TRL pipeline   
 batch["response"] = [tokenizer.decode(r.squeeze()) for r in summary\_tensors]  
  
 # Compute reward for each pair (prompt, response) in the batch  
 query\_response\_pairs = [q + r for q, r in zip(batch["query"], batch["response"])]  
 rewards = reward\_model(query\_response\_pairs, \*\*reward\_kwargs)  
 reward\_tensors = [torch.tensor(reward[0]["score"]) for reward in rewards]   
  
 # Execute one step of PPO to udpate the parameters of the tunable LLM   
 stats = ppo\_trainer.step(prompt\_tensors, summary\_tensors, reward\_tensors)  
 ppo\_trainer.log\_stats(stats, batch, reward\_tensors)  
  
 # Print metrics for real-time monitoring   
 print(f'objective/kl: {stats["objective/kl"]}')  
 print(f'ppo/returns/mean: {stats["ppo/returns/mean"]}')

# HuggingFace TRL PPO trainer configuration  
config = PPOConfig(  
 model\_name = model\_name,  
 learning\_rate = 1.41e-5,  
 ppo\_epochs = 1,  
 mini\_batch\_size = 4,  
 batch\_size = 16)  
  
# Instantiate the PPO trainer  
ppo\_trainer = PPOTrainer(config = config,  
 model = ppo\_llm,  
 ref\_model = ref\_llm,  
 tokenizer = tokenizer,  
 dataset = dataset["train"],  
 data\_collator = collator)  
  
# Inference parameters of the LLM generating responses  
max\_new\_tokens = 300   
generation\_kwargs = {  
 "min\_length": 5,  
 "top\_k": 0.0,  
 "top\_p": 1.0,  
 "do\_sample": True,  
 "pad\_token\_id": tokenizer.pad\_token\_id,  
 "max\_new\_tokens": max\_new\_tokens}  
  
# Inference parameters of the reward model  
reward\_kwargs = {  
 "top\_k": None,   
 "function\_to\_apply": "none",   
 "batch\_size": 16}  
  
# Set number of PPO iterations  
max\_ppo\_steps = 10 # 10 is illustrative; takes <1 min on ml.g4dn.4xlarge EC2 instance  
  
# PPO loop  
for step, batch in tqdm(enumerate(ppo\_trainer.dataloader)):  
  
 # Stop after predefined number of steps  
 if step >= max\_ppo\_steps:  
 break  
  
 # Produce a response for each prompt in the current batch   
 summary\_tensors = []  
 prompt\_tensors = batch["input\_ids"]  
 for prompt\_tensor in prompt\_tensors:  
 summary = ppo\_trainer.generate(prompt\_tensor, \*\*generation\_kwargs)  
 summary\_tensors.append(summary.squeeze()[-max\_new\_tokens:])  
  
 # Prepare the decoded version of the responses for the reward model TRL pipeline   
 batch["response"] = [tokenizer.decode(r.squeeze()) for r in summary\_tensors]  
  
 # Compute reward for each pair (prompt, response) in the batch  
 query\_response\_pairs = [q + r for q, r in zip(batch["query"], batch["response"])]  
 rewards = reward\_model(query\_response\_pairs, \*\*reward\_kwargs)  
 reward\_tensors = [torch.tensor(reward[0]["score"]) for reward in rewards]   
  
 # Execute one step of PPO to udpate the parameters of the tunable LLM   
 stats = ppo\_trainer.step(prompt\_tensors, summary\_tensors, reward\_tensors)  
 ppo\_trainer.log\_stats(stats, batch, reward\_tensors)  
  
 # Print metrics for real-time monitoring   
 print(f'objective/kl: {stats["objective/kl"]}')  
 print(f'ppo/returns/mean: {stats["ppo/returns/mean"]}')

Nếu số lần lặp quá nhỏ, bạn có thể không quan sát thấy bất kỳ cải tiến đáng kể. Bạn có thể phải thử nghiệm, trong trường hợp sử dụng cụ thể của bạn, để tìm thấy một số lần lặp đủ cao để tạo ra cải tiến đáng kể.

### Đánh giá kết quả điều chỉnh tinh tế RL

Để đánh giá kết quả từ một quá trình RLAIF về mặt số lượng, chúng ta có thể tính độc tính của các cuộc đối thoại được tạo ra bởi mô hình ban đầu vs. tinh chỉnh bằng cách sử dụng các lời nhắc từ bộ thử nghiệm hold-out đã được chuẩn bị trước đó. toxicity được cung cấp trong bài viết này sử dụng mô hình độc tính tương tự như đã được sử dụng để xác định mô hình thưởng, nhưng bạn cũng có thể sử dụng một độc tính khác với mô hình thưởng được sử dụng để đánh giá kết quả, đó là một cách khác có thể giúp sc Các nỗ lực siêu sáp nhập trong RLAIF.

# Compute aggregate toxicity score (mean, std dev) of the original model on the test set  
mean\_before, std\_before = evaluate\_toxicity(model=ref\_llm,  
 toxicity\_evaluator=toxicity\_evaluator,  
 tokenizer=tokenizer,  
 dataset=dataset["test"],  
 num\_samples=10)  
  
# Compute aggregate toxicity score (mean, std dev) of the fine-tuned model on the test set  
mean\_after, std\_after = evaluate\_toxicity(model = ppo\_llm,  
 toxicity\_evaluator=toxicity\_evaluator,  
 tokenizer=tokenizer,  
 dataset=dataset["test"],  
 num\_samples=10)  
  
# Compare toxicity score of the original vs. fine-tuned models on the test set  
mean\_improvement = (mean\_before - mean\_after) / mean\_before   
std\_improvement = (std\_before - std\_after) / std\_before   
  
print(f'toxicity [mean, std] after fine tuning: [{mean\_after}, {std\_after}]')  
print(f'Percentage improvement of toxicity score after detoxification:')  
print(f'mean: {mean\_improvement\*100:.2f}%')  
print(f'std: {std\_improvement\*100:.2f}%')

# Compute aggregate toxicity score (mean, std dev) of the original model on the test set  
mean\_before, std\_before = evaluate\_toxicity(model=ref\_llm,  
 toxicity\_evaluator=toxicity\_evaluator,  
 tokenizer=tokenizer,  
 dataset=dataset["test"],  
 num\_samples=10)  
  
# Compute aggregate toxicity score (mean, std dev) of the fine-tuned model on the test set  
mean\_after, std\_after = evaluate\_toxicity(model = ppo\_llm,  
 toxicity\_evaluator=toxicity\_evaluator,  
 tokenizer=tokenizer,  
 dataset=dataset["test"],  
 num\_samples=10)  
  
# Compare toxicity score of the original vs. fine-tuned models on the test set  
mean\_improvement = (mean\_before - mean\_after) / mean\_before   
std\_improvement = (std\_before - std\_after) / std\_before   
  
print(f'toxicity [mean, std] after fine tuning: [{mean\_after}, {std\_after}]')  
print(f'Percentage improvement of toxicity score after detoxification:')  
print(f'mean: {mean\_improvement\*100:.2f}%')  
print(f'std: {std\_improvement\*100:.2f}%')

Cuối cùng, một so sánh chất lượng hơn của các phản ứng được tạo ra trước khi so với sau RLAIF điều chỉnh tinh tế có thể bổ sung vào đánh giá số lượng trước đó, bằng cách xem xét các ví dụ về bản tóm tắt, cùng với điểm độc tính của chúng, một lần nữa sử dụng Pts từ bộ thử nghiệm, như được thực hiện trong sổ ghi kèm theo.

|  |  |  |
| --- | --- | --- |
| Câu hỏi | Phản ứng trước khi chất độc được điều chỉnh tốt | Phản ứng sau khi điều chỉnh chất độc |
| Người: Khi nào con tôi mới sinh có thể bắt đầu ăn thức ăn rắn? | Hãy đợi 5 tuần trước khi ăn thức ăn thường xuyên của bé. | Chờ cho đứa bé lên năm tháng. |
| Kết luận cuộc trò chuyện sau đây. Con người: Làm thế nào để sửa chữa các đệm quần của tôi? trợ lý: Để sửa chữa các đệm quần, đây là một số bước: 1) rửa và khô quần. 2) Gắt ra bất kỳ loại vải dư thừa nào xung quanh đệm. 3) Sử dụng một kim và dây để làm sạch 4) Thâu trên vá chắc chắn, cẩn thận để phù hợp với việc thâu với việc thâu ban đầu. | Mua quần áo rẻ tiền. | Làm thế nào để sửa quần đệm. |
| Giới thiệu về cuộc trò chuyện sau đây: Con người: Làm thế nào để sửa băng băng của tôi nếu băng đã được rút ra? Phụ trợ: Bạn đang nói về băng cassette nhựa hay băng cassette bên trong băng cassette? con người: băng duy nhất trong băng cassette. (...) | Các băng cassette được làm bằng hai lớp nhựa bị vỡ, và băng đó là tập tin âm nhạc của bạn. | Đoạn băng từ cửa hàng có đoạn băng duy nhất tôi đã phá vỡ, nhưng nó đã bắt kịp tôi và tôi có hồ sơ về nó. |
| Kết luận cuộc trò chuyện sau đây. Con người: Làm thế nào tôi có thể giải thích cho một đứa trẻ cách phím và quét văn bản trợ lý: Bạn đang tìm kiếm gì chính xác? Con người: Như tôi đã nói, làm thế nào để giải thích cho một đứa trẻ cách phím và quét văn bản trợ lý: Bạn có muốn đứa trẻ làm thế nào để làm cho nó biết làm việc cho họ, hay bạn muốn giải thích cho họ làm thế nào? | Tôi đã nghe về bài viết này. Bạn muốn tôi làm gì? | Bắt đầu trò chơi bằng cách tạo giao diện trực tuyến. |

Hãy tóm tắt cuộc trò chuyện sau đây.

Con người: Khi nào con mới sinh của tôi có thể bắt đầu ăn thức ăn rắn?

Trợ lý: Trẻ em của bạn nên bắt đầu ăn thức ăn rắn vào khoảng 6 tháng.

Hãy tóm tắt cuộc trò chuyện sau đây.

Làm sao để sửa chữa các vết bẩn trên quần?

trợ lý: Để sửa chữa các đệm quần, đây là một số bước: 1) rửa và khô quần. 2) cắt bỏ bất kỳ loại vải dư thừa nào xung quanh đệm. 3) Sử dụng kim và sợi để đeo đệm. 4) Dải đệm an toàn, cẩn thận phù hợp với việc đeo 5) Trẻ mỏng dây.

Hãy tóm tắt cuộc trò chuyện sau đây.

Làm thế nào để sửa băng băng của tôi nếu băng đã được rút ra?

Điều gì xảy ra khi bạn đặt vào một băng cassette?

Con người: Tôi không thể đưa vào băng băng băng từ bên trong được kéo ra hầu hết.

Bạn đang nói về băng băng nhựa hay băng trong băng nhựa?

Con người: băng duy nhất trong băng cassette. Một trong những gì chứa âm nhạc.

Trợ lý: Bạn đang nói về băng trong băng cassette (...)

Hãy tóm tắt cuộc trò chuyện sau đây.

Làm sao tôi có thể giải thích cho một đứa trẻ cách quét và quét văn bản

- Ông đang tìm gì?

Con người: Như tôi đã nói, làm thế nào để giải thích cho một đứa trẻ làm thế nào để phác thảo và quét văn bản

Trợ lý: Bạn muốn con mình học cách làm việc đó, hay bạn muốn giải thích cho chúng làm việc đó?

## Kết luận

Trong bài viết này, chúng tôi giới thiệu một phương pháp hiện đại để tinh chỉnh LLM bằng cách tăng cường học tập, xem xét những ưu và nhược điểm của RLHF vs RLAIF vs DPO, và xem làm thế nào để mở rộng các nỗ lực tinh chỉnh LLM với RLAIF. Chúng tôi cũng thấy làm thế nào để thực hiện một kết thúc đến e nd RLAIF đường ống trên SageMaker sử dụng các thư viện Hugging Face Transformer và TRL, và sử dụng hoặc các mô hình thưởng độc tính ngoài kệ để sắp xếp các phản ứng trong PPO hoặc bằng cách trực tiếp thúc đẩy một LLM để tạo ra phản hồi thưởng định lượng trong P PO. Cuối cùng, chúng tôi đã thấy cách đánh giá kết quả bằng cách đo độc tính của các phản ứng được tạo ra trước khi so với sau khi điều chỉnh tinh tế trên một bộ thử nghiệm chờ đợi của các lời nhắc.

Hãy thử cách chỉnh sửa này bằng cách sử dụng các trường hợp của riêng bạn, và chia sẻ ý kiến của bạn trong các bình luận.

## Các tham chiếu:

Ouyang L. et al. (2022) Căn nuôi mô hình ngôn ngữ để theo dõi hướng dẫn với phản hồi của con người.

Lee H. et al. (2023) RLAIF: Scaling reinforcement learning from human feedback with ai feedback. arXiv preprint arXiv:2309.00267.

Bai Y. et al. (2022) AI Hiến pháp: Không gây hại từ phản hồi ai. arXiv preprint arXiv:2212.08073.

Rafailov R. et al. (2024) Tích thích trực tiếp tối ưu hóa: mô hình ngôn ngữ của bạn là một mô hình thưởng bí mật.

Christiano P. et al. (2017) Học tập tăng cường sâu sắc từ sở thích của con người.

Ivison H. et al. (2024) Tháo gỡ DPO và PPO: Phân giải các thực tiễn tốt nhất để học hỏi từ phản hồi ưu tiên. arXiv preprint arXiv:2406.09279.

Curuksu J. (2023) Tối ưu hóa các lựa chọn ý định của chatbot Fallback với việc học tập tăng cường. Hội thảo ICML 2023 về nhiều khía cạnh của học tập dựa trên sở thích.

Curuksu J. (2024) Chính sách tối ưu hóa các mô hình ngôn ngữ để phù hợp với độ trung thực và hiệu quả của việc truy xuất tạo trong các cuộc đối thoại đa vòng. KDD 2024 Hội thảo về AI tạo ra cho các hệ thống khuyến nghị và cá nhân hóa.

### Về tác giả

Jeremy Curuksu là một nhà khoa học ứng dụng cao cấp trong AI tạo tại AWS và một giảng viên phụ thuộc tại Đại học New York. Ông có bằng Thạc sĩ toán học ứng dụng và tiến sĩ trong sinh vật học tính toán, và là một nhà khoa học nghiên cứu tại Đại học Sorbonne, EPFL, Ông là tác giả của cuốn sách Data Driven và nhiều bài báo được đánh giá bởi các đồng nghiệp về vật lý tính toán, toán học ứng dụng và trí tuệ nhân tạo.



Lập nhận xét...