TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

NGUYỄN VŨ THANH LIÊM - 52200206
TRẦN HỒ HOÀNG VŨ – 52200214
TRẦN KHIẾT LÔI – 52200216

BÁO CÁO CUỐI KỲ
NHẬP MÔN
XỬ LÝ NGÔN NGỮ TỰ NHIÊN

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2025

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

NGUYỄN VŨ THANH LIÊM - 52200206
TRẦN HỒ HOÀNG VŨ – 52200214
TRẦN KHIẾT LÔI – 52200216

BÁO CÁO CUỐI KỲ
NHẬP MÔN
XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Người hướng dẫn
PGS.TS. Lê Anh Cường

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2025

LỜI CẢM ƠN

Đầu tiên, chúng tôi xin gửi lời cảm ơn chân thành và tri ân sâu sắc đến **Ban giám hiệu Trường Đại học Tôn Đức Thắng** và **khoa Công nghệ thông tin** vì đã tạo điều kiện tối ưu về cơ sở vật chất và hệ thống thư viện hiện đại, đa dạng các loại tài liệu, giúp chúng tôi có thể dễ dàng và thuận tiện hơn trong việc tìm kiếm, nghiên cứu thông tin để hoàn thành bài báo cáo này.

Chúng tôi xin chân thành cảm ơn **PGS.TS. Lê Anh Cường**, giảng viên bộ môn, đã truyền đạt kiến thức một cách tận tình, chi tiết, định hướng tư duy, giúp chúng tôi có đầy đủ nền tảng để vận dụng tốt hơn vào việc thực hiện bài báo cáo này.

Mặc dù đã cố gắng vận dụng những kiến thức đã học được để hoàn thành bài báo cáo. Nhưng do kiến thức của chúng tôi còn hạn chế và không có nhiều kinh nghiệm thực tiễn nên khó tránh khỏi những thiếu sót trong quá trình nghiên cứu và trình bày. Do đó, chúng tôi rất mong nhận được sự góp ý của quý thầy cô để bài báo cáo của chúng tôi được hoàn thiện hơn.

Chúng tôi xin chân thành cảm ơn!

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Chúng tôi xin cam đoan đây là công trình nghiên cứu của riêng nhóm chúng tôi và được sự hướng dẫn khoa học của **PGS.TS. Lê Anh Cường**. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong báo cáo còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào chúng tôi xin hoàn toàn chịu trách nhiệm về nội dung báo cáo của mình. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 01 tháng 05 năm 2025

Tác giả

(Ký tên và ghi rõ họ tên)

Nguyễn Vũ Thanh Liêm

Trần Khiết Lôi

Trần Hồ Hoàng Vũ

TÓM TẮT

Bài báo cáo gồm 2 phần:

- Phần về Reinforcement Learning (RL):

  + Nội dung: Trình bày chi tiết các thuật toán RL nổi bật như PPO, DPO, TRPO và A2C, đồng thời phân tích rõ ràng ưu điểm, nhược điểm, và các ứng dụng thực tiễn của chúng trong mô hình ngôn ngữ lớn (LLMs), đặc biệt nhấn mạnh hiệu quả vượt trội của PPO và DPO.

  + Mục đích: Đánh giá một cách toàn diện hiệu quả của các thuật toán RL trong việc tinh chỉnh LLMs, từ đó hỗ trợ người đọc lựa chọn phương pháp tối ưu cho các bài toán phức tạp và đa dạng.

- Phần về mô hình dịch máy:

  + Nội dung: Đánh giá kỹ lưỡng hiệu suất của ba mô hình dịch máy quan trọng, bao gồm Transformer từ đầu, Transformer pretrained, và GPT-2 pretrained, dựa trên các chỉ số cụ thể từ bộ dữ liệu IWSLT15, kèm theo phân tích so sánh giữa chúng.

  + Mục đích: Cung cấp cái nhìn sâu sắc về hiệu quả của từng mô hình, nhấn mạnh lợi ích của pretrained weights, và hướng dẫn lựa chọn mô hình phù hợp cho nhiệm vụ dịch máy tiếng Anh - tiếng Việt.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

CHƯƠNG 1. CÁC THUẬT TOÁN REINFORCEMENT LEARNING TRONG MÔ HÌNH NGÔN NGỮ LỚN

1.1 Giới thiệu:

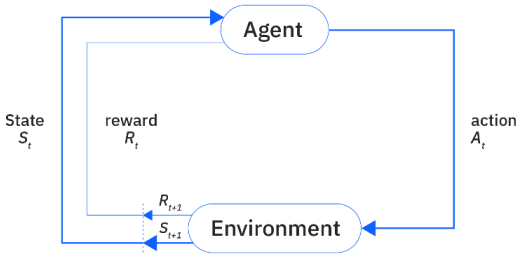
Reinforcement Learning (RL) là một nhánh của học máy, trong đó một tác nhân (agent) học cách đưa ra quyết định tối ưu bằng cách tương tác với môi trường (environment) và nhận thưởng (reward) dựa trên các hành động (action) của mình. Mục tiêu của RL là tối ưu hóa tổng thưởng tích lũy trong dài hạn thông qua việc thử nghiệm và học từ kinh nghiệm.

1.1.1 Quy trình Reinforcement Learning cơ bản:

Quy trình RL cơ bản được minh họa qua sơ đồ tương tác giữa Agent và Environment:

- Agent: Tác nhân đưa ra quyết định, trong bối cảnh LLMs có thể là mô hình ngôn ngữ.
- Environment: Môi trường mà agent tương tác, trong LLMs có thể là ngữ cảnh văn bản hoặc phản hồi người dùng.
- State (S_t): Trạng thái tại thời điểm t, đại diện cho thông tin hiện tại (ví dụ: prompt hoặc ngữ cảnh).
- Action (A_t): Hành động mà agent thực hiện (ví dụ: tạo một câu trả lời).
- Reward (R_t): Phản hồi từ môi trường, đánh giá chất lượng hành động (ví dụ: điểm số dựa trên độ chính xác của câu trả lời).
- Next State (S_{t+1}): Trạng thái mới sau khi thực hiện hành động.

Sơ đồ dưới đây mô tả quy trình này:



Hình 1. Quy trình tương tác giữa Agent và Environment trong RL

- Tại thời điểm t, agent nhận trạng thái S_t từ môi trường.
- Agent chọn hành động A_t dựa trên chính sách $\pi(A_t|S_t)$.
- Môi trường phản hồi với thưởng R_t và trạng thái mới S_{t+1}.
- Agent cập nhật chính sách hoặc hàm giá trị dựa trên thông tin nhận được.

Trong các mô hình ngôn ngữ lớn (LLMs), RL được sử dụng để tinh chỉnh (fine-tune) mô hình, giúp cải thiện chất lượng đầu ra, chẳng hạn như tạo văn bản chính xác, phù hợp ngữ cảnh, hoặc tuân thủ các tiêu chí đạo đức. Ví dụ, trong một hệ thống đối thoại, trạng thái S_t có thể là câu hỏi của người dùng, hành động A_t là câu trả lời được tạo, và thưởng R_t là đánh giá của người dùng về chất lượng câu trả lời.

1.1.2 Phân loại thuật toán Reinforcement Learning:

Các thuật toán RL được chia thành hai loại chính:

- Thuật toán dựa trên giá trị (Value-Based Algorithms):
  + Tối ưu hóa hàm giá trị hành động $Q(s,a)$ hoặc hàm giá trị trạng thái $V(s)$, ví dụ: SARSA và Deep Q-Learning (DQN).
  + Tuy nhiên, loại này ít phù hợp với LLMs do không gian hành động lớn.
- Thuật toán dựa trên chính sách (Policy-Based Algorithms):
  + Tối ưu hóa trực tiếp chính sách $\pi(a|s)$, phù hợp với LLMs nhờ khả năng xử lý không gian hành động liên tục và phức tạp.
  + Các thuật toán được trình bày trong báo cáo này (PPO, DPO, A2C, TRPO) đều thuộc nhóm này.

Phần sau trình bày chi tiết từng thuật toán, bao gồm công thức toán học, giải thuật, ý nghĩa, ứng dụng trong LLMs, và so sánh giữa các thuật toán.

1.2 Các thuật toán Reinforcement Learning trong LLMs:

1.2.1 Proximal Policy Optimization (PPO):

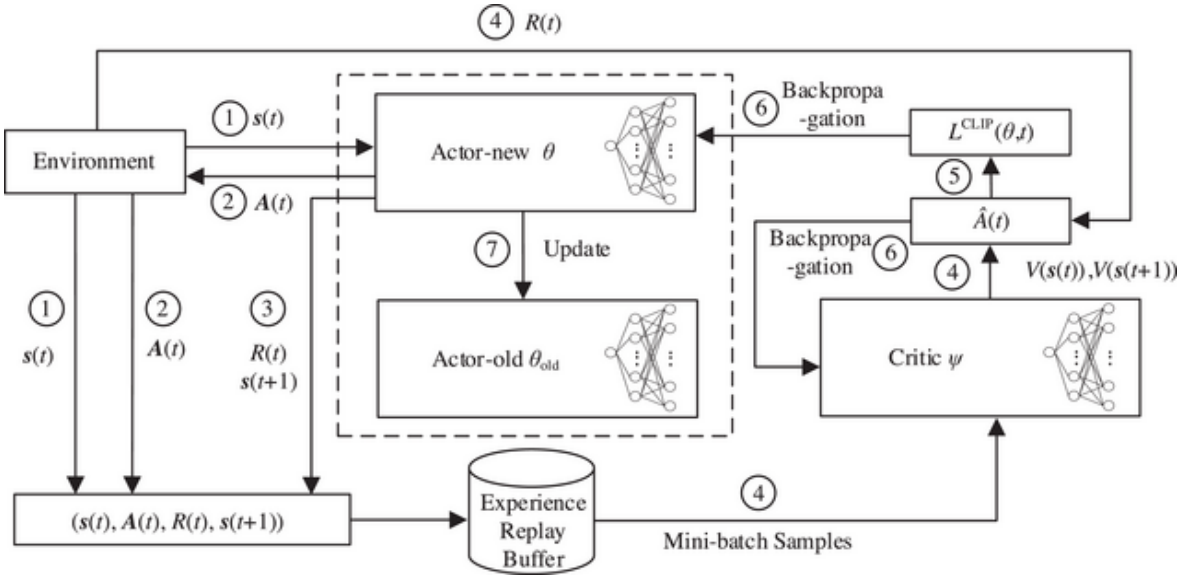
1.2.1.1 Tổng quan:

Proximal Policy Optimization (PPO) là một thuật toán học tăng cường (Reinforcement Learning) thuộc nhóm on-policy, được đề xuất bởi Schulman et al. (2017).

PPO nổi bật nhờ khả năng cân bằng giữa hiệu quả tối ưu hóa chính sách và sự ổn định trong huấn luyện, đặc biệt phù hợp với các bài toán có không gian hành động lớn như tinh chỉnh mô hình ngôn ngữ lớn (LLMs), ví dụ: InstructGPT và ChatGPT.

Điểm đặc trưng của PPO là sử dụng cơ chế clipping (giới hạn thay đổi chính sách) để tránh các cập nhật quá lớn, bảo vệ kiến thức đã học và cải thiện độ tin cậy của đầu ra.

1.2.1.2 Cơ chế hoạt động:



Hình 2. Cơ chế hoạt động của thuật toán PPO

PPO hoạt động thông qua sự tương tác giữa tác nhân (agent) và môi trường, sử dụng hai mô hình chính: Actor (điều khiển chính sách) và Critic (đánh giá giá trị trạng thái), cùng với cơ chế lưu trữ kinh nghiệm và tối ưu hóa có giới hạn. Quy trình này được minh họa qua các bước sau:

- Nhận trạng thái từ môi trường: Tác nhân nhận trạng thái hiện tại s_t từ môi trường (Environment).

- Chọn hành động: Actor-new (θ) sử dụng chính sách hiện tại để chọn hành động $A(t)$ dựa trên trạng thái $s(t)$.
- Nhận phần thưởng và trạng thái mới: Môi trường phản hồi với phần thưởng $R(t)$ và và trạng thái tiếp theo $s(t+1)$.
- Lưu trữ kinh nghiệm:

  + Bộ tứ $\big(s(t), A(t), R(t), s(t+1)\big)$ được lưu vào Experience Replay Buffer.

  + Critic (ψ) sử dụng các mẫu mini-batch từ buffer để tính giá trị trạng thái $V\big(s(t)\big)$ và $V\big(s(t+1)\big)$, từ đó ước lượng lợi thế $\hat{A}(t)$.
- Tính lợi thế (Advantage): Lợi thế $\hat{A}(t)$ được tính dựa trên giá trị trạng thái từ Critic, đo lường mức độ tốt hơn của hành động $A(t)$ so với giá trị trung bình tại trạng thái $s(t)$.
- Tối ưu hóa chính sách với clipping:

  + Actor-new (θ) tính tỷ lệ chính sách $r_t(\theta) = \frac{\pi_\theta\big(A(t)|s(t)\big)}{\pi_{\theta_{old}}\big(A(t)|s(t)\big)}$ so với Actor-old (θ_{old}).

  + Hàm mất mát clipping $L^{CLIP}(\theta)$ được tối ưu hóa bằng backpropagation, giới hạn $r_t(\theta)$ trong khoảng $[1-\epsilon, 1+\epsilon]$ để đảm bảo ổn định.
- Cập nhật mô hình: Tham số của Actor-new (θ) được cập nhật, và Actor-old (θ_{old}) được đồng bộ hóa với Actor-new để chuẩn bị cho vòng lặp tiếp theo.

Cơ chế này đảm bảo PPO cải thiện chính sách một cách ổn định, sử dụng dữ liệu từ Experience Replay Buffer và cơ chế clipping để tránh các cập nhật quá lớn, phù hợp với các bài toán phức tạp như tinh chỉnh LLMs.

1.2.1.3 Công thức:

PPO tối ưu hóa chính sách thông qua một hàm mục tiêu kết hợp nhiều thành phần. Dưới đây là các công thức chính với chú thích đầy đủ cho các biến:

- Hàm mục tiêu chính (Clipped Surrogate Objective):

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min\left(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right]$$

- Hàm mất mát tổng quát:

$$L(\theta) = \hat{\mathbb{E}}_t[L^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)]$$

Trong đó:

- $L^{CLIP}(\theta)$: Hàm mục tiêu chính của PPO, được tối ưu hóa để cập nhật chính sách, với θ là tham số của chính sách hiện tại.
- $\hat{\mathbb{E}}_t$: Ký hiệu kỳ vọng, tính trung bình trên các mẫu dữ liệu tại thời điểm t, dựa trên quỹ đạo thu thập từ chính sách cũ.
- $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$: Tỷ lệ xác suất giữa chính sách mới (π_θ) và chính sách cũ ($\pi_{\theta_{old}}$), phản ánh sự thay đổi của chính sách.
  + $\pi_\theta(a_t|s_t)$: Xác suất chọn hành động a_t trong trạng thái s_t theo chính sách mới θ.
  + $\pi_{\theta_{old}}(a_t|s_t)$: Xác suất chọn hành động a_t trong trạng thái s_t theo chính sách cũ θ_{old}.
- $\hat{A}_t$: Ước lượng lợi thế (advantage), đo lường mức độ tốt hơn của hành động a_t, so với giá trị trung bình tại trạng thái s_t, thường được tính bằng Generalized Advantage Estimation (GAE).
- $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)$: Hàm giới hạn giá trị $r_t(\theta)$ trong khoảng $[1-\epsilon, 1+\epsilon]$, nhằm tránh thay đổi chính sách quá lớn.
- ϵ: Hằng số clip (thường 0.1-0.2), giới hạn mức độ thay đổi chính sách.
- $L(\theta)$: Hàm mất mát tổng quát của PPO, kết hợp các thành phần để tối ưu hóa chính sách.
- $L^{VF}(\theta) = (V_\theta(s_t) - R_t)^2$: Hàm mất mát giá trị, đo lường sai số giữa giá trị dự đoán và phần thưởng thực tế.
  + $V_\theta(s_t)$: Hàm giá trị trạng thái, ước lượng phần thưởng kỳ vọng từ trạng thái s_t, được tính bởi mạng Critic với tham số θ.

+ R_t: Phần thưởng chiết khấu tích lũy từ thời điểm t, thường tính bằng $R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots$, với r_t là phần thưởng tại bước t và γ là hệ số chiết khấu (thường 0.9-0.99).

- $S[\pi_\theta](s_t)$: Entropy của chính sách π_θ tại trạng thái s_t, tính bằng $S[\pi_\theta](s_t) = -\sum_a \pi_\theta(a|s_t) \log \pi_\theta(a|s_t)$, khuyến khích sự đa dạng trong hành động để tránh hội tụ sớm vào một chính sách cục bộ..
- c_1: Hệ số cân bằng cho hàm mất mát giá trị, thường đặt khoảng 0.5.
- c_2: Hệ số cân bằng cho entropy, thường đặt nhỏ (ví dụ: 0.01), để điều chỉnh mức độ khuyến khích đa dạng hành động.

1.2.1.4 Giải thuật:

a. Khởi tạo: Tải mô hình ngôn ngữ với tham số chính sách θ (Actor-new) và tham số giá trị ψ (Critic). Đồng thời, khởi tạo Actor-old $\theta_{old} = \theta$ để lưu chính sách cũ.
b. Thu thập dữ liệu:
   - Tạo quỹ đạo (trajectories) bằng cách cho tác nhân tương tác với môi trường dựa trên chính sách hiện tại π_θ.
   - Lưu trữ các bộ tứ (s_t, a_t, r_t, s_{t+1}) vào Experience Replay Buffer. Trong đó: s_t là trạng thái, a_t là hành động, r_t là phần thường và s_{t+1} là trạng thái tiếp theo.
   - Tính phần thưởng chiết khấu $R_t = \sum_{k=t}^{T} \gamma^{k-t} r_k$ cho mỗi bước, với γ là hệ số chiết khấu.
c. Tính lợi thế:
   - Sử dụng Critic (ψ) để ước lượng giá trị trạng thái $V(s_t)$ và $V(s_{t+1})$.
   - Sử dụng Generalized Advantage Estimation (GAE) để tính $\hat{A}_t$, dựa trên giá trị trạng thái và phần thưởng r_t.
d. Tính tỉ lệ chính sách:

- Tính $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ cho từng mẫu trong mini-batch từ Experience Replay Buffer.

e. Tối ưu hóa chính sách:
   - Tối ưu hóa hàm mất mát $L^{CLIP}(\theta)$ với cơ chế clipping, giới hạn $r_t(\theta)$ trong $[1-\epsilon, 1+\epsilon]$.
   - Tối ưu hóa $L(\theta)$ bằng gradient ascent, thường trong 5-10 bước trên dữ liệu mini-batch, sử dụng bộ tối ưu hóa như Adam với tốc độ học nhỏ (ví dụ: 0.0001).

f. Tối ưu hóa hàm giá trị:
   - Tính hàm mất mát giá trị L_t^{VF} dựa trên sai số $V_\psi(s_t)$ và R_t.
   - Tối ưu hóa L_t^{VF} bằng gradient descent để cập nhật tham số ψ của Critic.

g. Cập nhật Actor-old: Gán $\theta_{old} \leftarrow \theta$ để đồng bộ Actor-old với Actor-new.

h. Lặp lại: Quay lại bước b cho đến khi mô hình hội tụ hoặc đạt số epoch mong muốn.

1.2.1.5 Ý nghĩa:

- Ổn định: Cơ chế clipping giới hạn tỷ lệ $r_t(\theta)$ trong khoảng $[1-\epsilon, 1+\epsilon]$, ngăn chặn cập nhật chính sách quá lớn, giúp duy trì hiệu suất và bảo vệ kiến thức đã học.
- Hiệu quả: Kết hợp lợi thế $\hat{A}_t$ và entropy $S[\pi_\theta]$ giúp PPO hội tụ nhanh hơn và tránh bị kẹt ở các điểm tối ưu cục bộ.
- Ứng dụng trong LLMs: PPO cải thiện tính hữu ích, trung thực và an toàn của văn bản được sinh ra, đặc biệt khi tinh chỉnh dựa trên phản hồi người dùng hoặc sở thích.

1.2.1.6 Ứng dụng trong LLMs:

PPO được sử dụng rộng rãi để tinh chỉnh các mô hình như InstructGPT, ChatGPT, và các biến thể của Llama.

Trong các hệ thống đối thoại, PPO điều chỉnh mô hình để tạo câu trả lời tự nhiên, chính xác và tuân thủ các tiêu chí đạo đức, dựa trên phần thưởng được định lượng từ đánh giá con người hoặc các tiêu chí định sẵn.

1.2.2 Direct Preference Optimization (DPO):

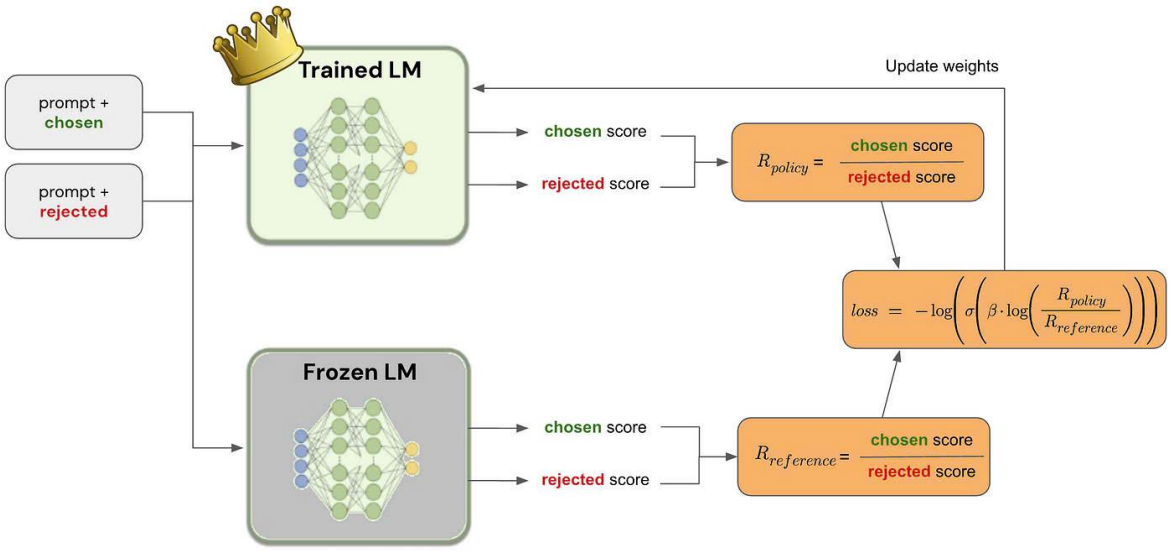
1.2.2.1 Tổng quan:

Direct Preference Optimization (DPO) là một thuật toán học tăng cường (Reinforcement Learning) on-policy, được đề xuất bởi Rafailov et al. (2023).

DPO được thiết kế để tối ưu hóa chính sách trực tiếp dựa trên dữ liệu sở thích (preference data) từ người dùng, loại bỏ nhu cầu sử dụng mô hình thưởng riêng biệt như trong các phương pháp truyền thống (ví dụ: PPO).

Phương pháp này sử dụng một mô hình ngôn ngữ cố định (Frozen LM) làm tham chiếu và so sánh với mô hình đang được huấn luyện (Trained LM), giúp đơn giản hóa quy trình tinh chỉnh và cải thiện hiệu quả trong các mô hình ngôn ngữ lớn (LLMs) như Llama và Mistral.

1.2.2.2 Cơ chế hoạt động:



Hình 3. Cơ chế hoạt động thuật toán DPO

DPO hoạt động bằng cách so sánh hai đầu ra từ cùng một ngữ cảnh (prompt): một đầu ra được ưu tiên (chosen response) và một đầu ra không được ưu tiên (rejected response). Quy trình này được minh họa qua hai mô hình chính:

- Trained LM: Mô hình đang được huấn luyện, với trọng số được cập nhật dựa trên dữ liệu sở thích.
- Frozen LM: Mô hình tham chiếu cố định, thường là phiên bản ban đầu của mô hình, không thay đổi trong quá trình huấn luyện.

Quy trình cụ thể:

- Input: Mỗi ngữ cảnh (prompt) được kết hợp với hai đầu ra: prompt + chosen (được ưa thích) và prompt + rejected (bị từ chối).
- Scores: Cả Trained LM và Frozen LM tính toán chosen score và rejected score dựa trên xác suất sinh ra các đầu ra này.
- Tỷ lệ chính sách (R_{policy}): Được tính bằng tỉ lệ giữa chosen score và rejected score từ Trained LM.
- Tỷ lệ tham chiếu ($R_{reference}$): Được tính tương tự từ Frozen LM.
- Hàm mất mát: DPO sử dụng sự khác biệt giữa xác suất từ Trained LM và Frozen LM để tối ưu hóa chính sách, ưu tiên các đầu ra được ưa thích.

1.2.2.3 Công thức:

Hàm mất mát:

$$L_{DPO}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{ref}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{ref}(y_l|x)}\right)\right]$$

Trong đó:

- $L_{DPO}(\theta)$: Hàm mất mát của DPO, tối ưu hóa chính sách π_θ.
- $\mathbb{E}_{(x,y_w,y_l)\sim D}$: Ký hiệu kỳ vọng, tính trên tập dữ liệu D chứa các cặp (x, y_w, y_l).
- x: Ngữ cảnh (prompt).
- y_w: Đầu ra được ưu tiên (winning/choosen response), được người dùng đánh giá cao hơn.

- y_l: Đầu ra không được ưu tiên (losing/rejected response).
- $\pi_\theta(y|x)$: Xác suất sinh ra đầu ra y từ ngữ cảnh x theo chính sách hiện tại (Trained LM).
- $\pi_{ref}(y|x)$: Xác suất sinh ra y từ x theo chính sách tham chiếu (Frozen LM), thường là mô hình ban đầu.
- β: Hệ số điều chỉnh, thường 0.1-1.0, kiểm soát mức độ phạt khi mô hình dự đoán sai sở thích.
- σ: Hàm sigmoid, $\sigma(z) = \frac{1}{1+e^{-z}}$, chuyển đổi giá trị thành xác suất.

Hàm mất mát này khuyến khích chính sách hiện tại (π_θ) tăng xác suất cho y_w (choosen response) và giảm xác suất cho y_l (rejected response) so với chính sách tham chiếu (π_{ref}).

1.2.2.4 Giải thuật:

a. Khởi tạo: Tải mô hình ngôn ngữ ban đầu làm Frozen LM (π_{ref}) và khởi tạo Trained LM (π_θ) với cùng kiến trúc, nhưng trọng số sẽ được cập nhật..
b. Thu thập dữ liệu sở thích: Thu thập các cặp (x, y_w, y_l) dựa trên phản hồi hoặc đánh giá từ con người.
c. Tính xác suất:
   - Trained LM (π_θ) và Frozen LM (π_{ref}) tính xác suất $\pi_\theta(y_w|x)$, $\pi_\theta(y_l|x)$, $\pi_{ref}(y_w|x)$ và $\pi_{ref}(y_l|x)$.
   - Sử dụng các xác suất này để tính toán hàm mất mát.
d. Tối ưu hóa: Sử dụng gradient descent để giảm thiểu hàm mất mát, cập nhật trọng số của Trained LM (π_θ).
e. Lặp lại: Quay lại bước b đến d cho đến khi mô hình hội từ hoặc đạt số epoch mong muốn.

1.2.2.5 Ý nghĩa:

- Đơn giản hóa quy trình: DPO không cần mô hình thưởng riêng, giảm độ phức tạp và tài nguyên tính toán so với PPO.

- Tập trung vào sở thích: Trực tiếp sử dụng dữ liệu sở thích, phù hợp cho các ứng dụng cần phản hồi con người (ví dụ: hệ thống đối thoại).
- Ổn định: Frozen LM đóng vai trò tham chiếu cố định, giúp duy trì tính nhất quán trong huấn luyện.
- Hạn chế: Hiệu quả của DPO phụ thuộc vào chất lượng dữ liệu sở thích; nếu dữ liệu không đủ đa dạng hoặc phong phú, mô hình có thể không đạt hiệu suất tối ưu.

1.2.2.6 Ứng dụng trong LLMs:

DPO được sử dụng rộng rãi để tinh chỉnh các mô hình như Llama, Mistral, và các biến thể của GPT, đặc biệt trong các tác vụ yêu cầu cá nhân hóa hoặc tuân thủ tiêu chí cụ thể.

Ví dụ, trong hệ thống đối thoại, DPO có thể điều chỉnh mô hình để ưu tiên các câu trả lời thân thiện, an toàn, và phù hợp với ngữ cảnh văn hóa dựa trên sở thích từ người dùng.

1.2.3 Trust Region Policy Optimization (TRPO):

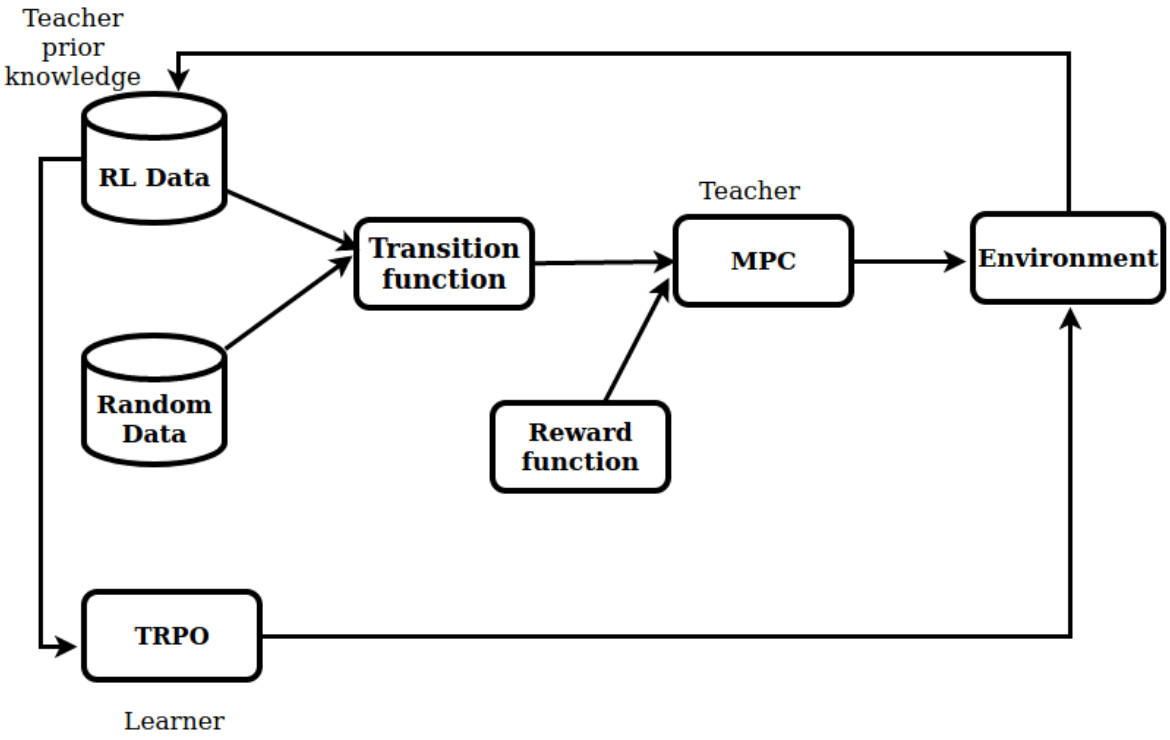
1.2.3.1 Tổng quan:

Trust Region Policy Optimization (TRPO) là một thuật toán học tăng cường (Reinforcement Learning) thuộc nhóm on-policy, được đề xuất bởi Schulman et al. (2015).

TRPO nổi bật nhờ việc đảm bảo cập nhật chính sách nằm trong một vùng tin cậy (trust region) bằng cách sử dụng độ đo Kullback-Leibler (KL) để giới hạn sự thay đổi chính sách, mang lại sự ổn định vượt trội so với các phương pháp đơn giản.

Đây là tiền thân của PPO và phù hợp với các bài toán có không gian hành động lớn, bao gồm tinh chỉnh mô hình ngôn ngữ lớn (LLMs).

1.2.3.2 Cơ chế hoạt động:



Hình 4. Cơ chế hoạt động thuật toán TRPO

TRPO hoạt động thông qua một quy trình tích hợp chặt chẽ giữa việc thu thập dữ liệu từ môi trường, sử dụng kiến thức từ một mô hình dự đoán (Teacher MPC), và tối ưu hóa chính sách trong một vùng an toàn. Hình minh họa cho thấy:

- Dữ liệu đầu vào: TRPO nhận dữ liệu từ hai nguồn chính: "RL Data" (dữ liệu thu thập từ tương tác thực tế) và "Random Data" (dữ liệu ngẫu nhiên để tăng đa dạng).
- Transition function: Chuyển đổi trạng thái và hành động thành trạng thái tiếp theo, được hỗ trợ bởi Teacher MPC để dự đoán các bước tiếp theo.
- Reward function: Đánh giá chất lượng hành động dựa trên phản hồi từ môi trường.
- Teacher MPC: Một mô hình dự đoán (Model Predictive Control) cung cấp kiến thức trước (prior knowledge) để hướng dẫn quá trình học.
- Môi trường (Environment): Tương tác trực tiếp với TRPO để cung cấp phản hồi thực tế.

- Learner (TRPO): Cập nhật chính sách dựa trên dữ liệu thu thập được, đảm bảo các thay đổi nằm trong vùng tin cậy.

Quy trình chi tiết bao gồm:

- Thu thập dữ liệu: Tác nhân tương tác với môi trường dựa trên chính sách hiện tại, sử dụng dữ liệu từ RL Data và hỗ trợ từ Teacher MPC.
- Tính toán lợi thế: Ước lượng mức độ tốt của hành động so với giá trị trung bình.
- Tối ưu hóa chính sách: Sử dụng phương pháp conjugate gradient để tối ưu hóa hàm mục tiêu với ràng buộc trust region.
- Cập nhật và lặp lại: Đồng bộ hóa chính sách và tiếp tục quá trình.

Cơ chế này đảm bảo TRPO cân bằng giữa việc cải thiện hiệu suất và duy trì sự ổn định, đặc biệt khi tích hợp với dữ liệu phong phú từ môi trường và mô hình dự đoán.

1.2.3.3 Công thức:

TRPO tối ưu hóa chính sách thông qua một hàm mục tiêu với ràng buộc vùng tin cậy. Các công thức chính được cập nhật như sau:

Hàm mục tiêu:

$$L(\theta) = \mathbb{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right]$$

Ràng buộc trust region:

$$\mathbb{E}_t \left[D_{KL} \left(\pi_{\theta_{old}}(a_t|s_t) \parallel \pi_\theta(a_t|s_t) \right) \right] \leq \delta$$

Trong đó:

- $L(\theta)$: Hàm mục tiêu, tối ưu hóa chính sách mới π_θ.
- $\mathbb{E}_t$: Ký hiệu kỳ vọng, tính trên các mẫu tại thời điểm t.
- $\pi_\theta(a_t|s_t)$: Xác suất chọn hành động a_t trong trạng thái s_t theo chính sách mới.
- $\pi_{\theta_{old}}(a_t|s_t)$: Xác suất chọn hành động a_t trong trạng thái s_t theo chính sách cũ.

- $\hat{A}_t$: Ước lượng lợi thế tại thời điểm t, đo lường mức độ tốt của hành động a_t so với giá trị trung bình.
- D_{KL}: Độ đo Kullback-Leibler, đo sự khác biệt giữa chính sách cũ và mới.
- δ: Ngưỡng trust region, giới hạn mức độ thay đổi chính sách (thường được đặt nhỏ, ví dụ: 0.01).

Hàm mục tiêu được tối ưu hóa bằng phương pháp conjugate gradient, đảm bảo sự ổn định trong các bước cập nhật.

1.2.3.4 Giải thuật:

a. Khởi tạo: Tải mô hình ngôn ngữ với chính sách ban đầu θ và ngưỡng trust region δ.
b. Thu thập dữ liệu:
   - Tạo quỹ đạo (trajectories) bằng cách cho tác nhân tương tác với môi trường, hỗ trợ bởi Teacher MPC và dữ liệu từ RL Data/Random Data.
   - Lưu trữ các bộ tứ (s_t, a_t, r_t, s_{t+1}) (trạng thái, hành động, phần thưởng, trạng thái tiếp theo).
c. Tính lợi thế: Ước lượng $\hat{A}_t$ dựa trên giá trị trạng thái và phần thưởng từ môi trường.
d. Cập nhật chính sách:
   - Tối ưu hóa $L(\theta)$ với ràng buộc trust region bằng conjugate gradient.
   - Đảm bảo $\mathbb{E}_t[D_{KL}] \leq \delta$
e. Đồng bộ hóa: Gán $\theta_{old} \leftarrow \theta$ cho vòng lặp tiếp theo.
f. Lặp lại: Quay lại bước b cho đến khi mô hình hội tụ hoặc đạt số epoch mong muốn.

1.2.3.5 Ý nghĩa:

- Ổn định: Ràng buộc trust region hạn chế thay đổi chính sách quá lớn, bảo vệ kiến thức đã học và tránh mất ổn định trong huấn luyện.
- Hiệu quả: Tích hợp với Teacher MPC và dữ liệu đa dạng (RL Data, Random Data) giúp TRPO xử lý tốt các bài toán phức tạp.

- Ứng dụng trong LLMs: TRPO phù hợp để tinh chỉnh mô hình ngôn ngữ, đặc biệt trong các nghiên cứu ban đầu, tối ưu hóa đầu ra theo tiêu chí cụ thể dựa trên phản hồi từ môi trường.

1.2.3.6 Ứng dụng trong LLMs:

TRPO được sử dụng trong tinh chỉnh mô hình ngôn ngữ, đặc biệt trong các nghiên cứu ban đầu về RL cho LLMs.

Ví dụ, nó có thể tối ưu hóa đầu ra theo các tiêu chí cụ thể (như tính chính xác hoặc tuân thủ đạo đức) bằng cách tận dụng dữ liệu phong phú từ môi trường và sự hướng dẫn của Teacher MPC.

1.2.4 Advantage Actor-Critic (A2C):

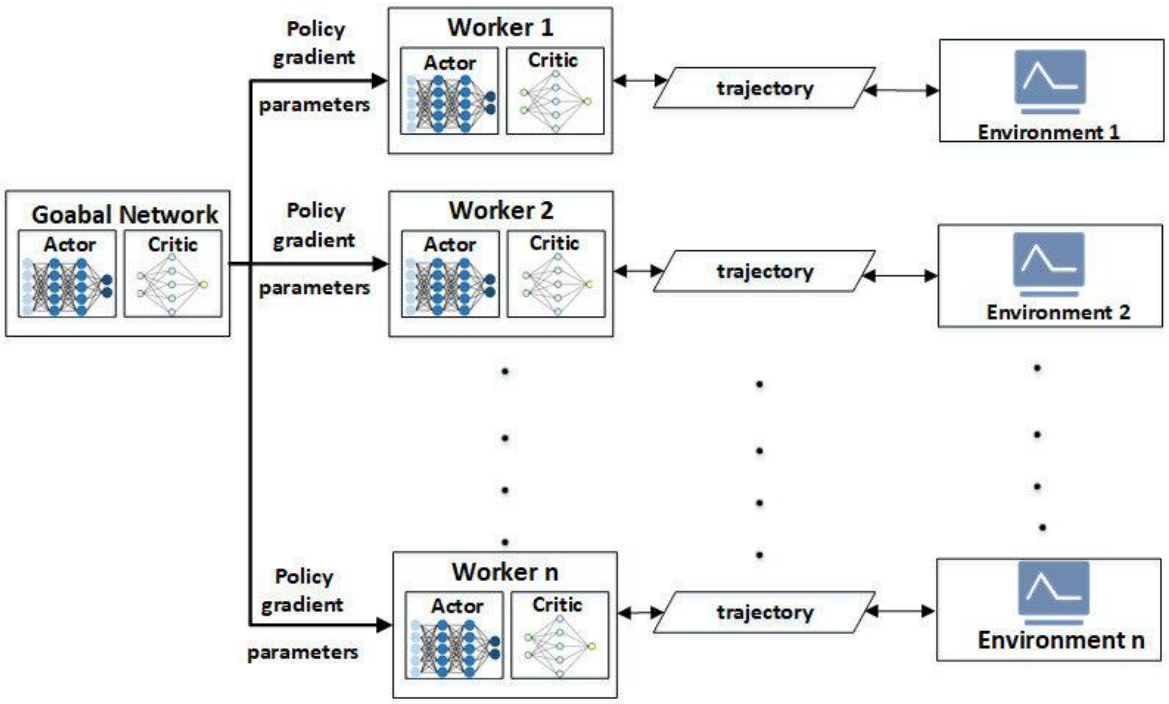
1.2.4.1 Tổng quan:

Advantage Actor-Critic (A2C) là một thuật toán Reinforcement Learning (RL) on-policy, kết hợp phương pháp dựa trên chính sách (actor) và dựa trên giá trị (critic) để tối ưu hóa hiệu suất của tác nhân.

Được phát triển từ Actor-Critic, A2C cải thiện hiệu quả bằng cách sử dụng nhiều tác nhân song song (workers) để thu thập dữ liệu từ các môi trường độc lập, sau đó tổng hợp thông tin để cập nhật một mạng toàn cục (global network).

Điều này giúp giảm phương sai và tăng tốc độ hội tụ, đặc biệt phù hợp với việc tinh chỉnh các mô hình ngôn ngữ lớn (LLMs).

1.2.4.2 Cơ chế hoạt động:



Hình 5. Cơ chế hoạt động thuật toán A2C

A2C hoạt động dựa trên sự phối hợp giữa một Global Network và nhiều Worker, như được minh họa trong hình 4. Quy trình chi tiết bao gồm:

- Global Network: Chứa các tham số chung cho cả Actor $(\pi(s,a)$ và Critic $(V(s))$, đóng vai trò trung tâm để cập nhật chính sách và giá trị trạng thái.
- Workers: Các tác nhân song song, mỗi cái tương tác với một bản sao môi trường riêng lẻ, thu thập quỹ đạo (trajectories) gồm trạng thái, hành động, và phần thưởng.
- Tương tác môi trường: Mỗi Worker chọn hành động dựa trên chính sách từ Global Network, nhận phản hồi (phần thưởng và trạng thái mới) từ môi trường, và gửi dữ liệu về Global Network.
- Cập nhật đồng bộ: Sau khi thu thập dữ liệu từ tất cả các Worker, Global Network tính toán gradient dựa trên lợi thế (advantage) và cập nhật tham số, sau đó đồng bộ hóa lại với các Worker.

Cơ chế này tận dụng tính song song để tăng cường tính đa dạng của dữ liệu và giảm thời gian tính toán, đặc biệt hiệu quả trong các bài toán phức tạp như tinh chỉnh LLMs.

1.2.4.3 Công thức:

Hàm mất mát của A2C được tối ưu hóa cho cả Actor và Critic:

- Hàm mất mát của actor:

$$L_{actor} = -\mathbb{E}_{s,a\sim\rho(\tau)}[\log \pi(a|s;\theta) \cdot A(s,a)]$$

- Hàm mất mát của critic:

$$L_{critic} = \mathbb{E}_{s\sim\rho(\tau)}\left[\left(V(s;\phi) - \left(R + \gamma V(s';\phi)\right)\right)^2\right]$$

Trong đó:

- L_{actor}: Hàm mất mát của actor, được tối ưu hóa để cập nhật chính sách π nhằm tăng cường xác suất chọn các hành động có lợi thế cao.
- $\mathbb{E}_{s,a\sim\rho(\tau)}$: Kỳ vọng (trung bình) được tính trên các cặp trạng thái s và hành động a thu thập từ phân phối quỹ đạo $\rho(\tau)$, trong đó $\rho(\tau)$ là phân phối xác suất của các quỹ đạo $\tau = (s_0, a_0, r_0, s_1, \dots)$ được tạo ra bởi các Worker song song tương tác với môi trường.
- $\log \pi(a|s;\theta)$: Logarit của xác suất chọn hành động a trong trạng thái s theo chính sách mới π với tham số θ (trọng số của mạng Actor).
- $A(s,a)$: Lợi thế (advantage), đo lường mức độ tốt hơn của hành động a trong trạng thái s so với giá trị trung bình, được tính bằng $A(s,a) = Q(s,a) - V(s)$, trong đó $Q(s,a)$ là giá trị hành động (ước lượng phần thưởng kỳ vọng khi thực hiện a trong s) và $V(s)$ là giá trị trạng thái (ước lượng phần thưởng kỳ vọng từ s).
- L_{critic}: Hàm mất mát của critic, được tối ưu hóa để giảm sai số giữa giá trị trạng thái dự đoán và giá trị thực tế.
- $\mathbb{E}_{s\sim\rho(\tau)}$: Kỳ vọng được tính trên các trạng thái s từ phân phối quỹ đạo $\rho(\tau)$, phản ảnh dữ liệu thu thập từ các Worker.
- $V(s;\phi)$: Giá trị trạng thái dự đoán tại trạng thái s, được tính bởi mạng Critic với tham số ϕ (trọng số của mạng Critic).
- R_t: Phần thưởng tức thời nhận được tại bước thời gian t từ môi trường sau khi thực hiện hành động a.

- γ: Hệ số chiết khấu (discount factor), nằm trong khoảng $(0,1]$ (thường 0.9-0.99), quyết định tầm quan trọng của thưởng tương lai so với phần thưởng hiện tại.
- $V(s';\phi)$: Giá trị trạng thái dự đoán tại trạng thái tiếp theo s', cũng được tính bởi mạng Critic với cùng tham số ϕ.
- $R + \gamma V(s';\phi)$: Phần thưởng chiết khấu mục tiêu, đại diện cho tổng phần thưởng kỳ vọng từ bước hiện tại, bao gồm phần thưởng tức thời R_t và giá trị trạng thái tương lai chiết khấu $\gamma V(s')$.

1.2.4.4 Giải thuật:

a. Khởi tạo:
   - Tải Global Network với tham số θ (Actor) và ϕ (Critic).
   - Khởi tạo nhiều Worker, mỗi cái kết nối với một môi trường độc lập.

b. Thu thập dữ liệu:
   - Mỗi Worker lấy chính sách từ Global Network, tương tác với môi trường để tạo quỹ đạo (trajectories) gồm (s, a, r, s').
   - Thu thập dữ liệu từ tất cả các Worker trong một khoảng thời gian hoặc số bước nhất định.

c. Tính lợi thế:
   - Sử dụng Critic trong Global Network để ước lượng $V(s)$ và $V(s')$.
   - Tính $A(s, a)$ dựa trên giá trị phần thưởng chiết khấu và chuẩn hóa để giảm phương sai.

d. Cập nhật actor:
   - Tối ưu hóa L_{actor} bằng gradient ascent, sử dụng dữ liệu từ tất cả các Worker.

e. Cập nhật critic:
   - Tối ưu hóa L_{critic} bằng gradient descent để cải thiện ước lượng giá trị trạng thái..

f. Đồng bộ hóa: Cập nhật tham số của Global Network và truyền lại cho tất cả Worker.
g. Lặp lại: Quay lại bước b cho đến khi mô hình hội tụ hoặc đạt số epoch mong muốn.

1.2.4.5 Ý nghĩa:

- Hiệu quả nhờ song song hóa: Việc sử dụng nhiều Worker giúp thu thập dữ liệu đa dạng và nhanh chóng, giảm thời gian huấn luyện so với các phương pháp tuần tự.
- Giảm phương sai: Lợi thế $A(s, a)$ kết hợp với dữ liệu từ nhiều môi trường giúp cải thiện độ ổn định của gradient chính sách.
- Ứng dụng trong LLMs: A2C phù hợp để tinh chỉnh mô hình ngôn ngữ trong các hệ thống đối thoại hoặc tạo văn bản, nơi cần phản hồi nhanh chóng và chính xác từ nhiều ngữ cảnh.

1.2.4.6 Ứng dụng trong LLMs:

A2C được áp dụng để tinh chỉnh mô hình ngôn ngữ, đặc biệt trong các hệ thống yêu cầu tương tác đa ngữ cảnh.

Ví dụ, trong một chatbot, A2C có thể sử dụng nhiều Worker để mô phỏng các cuộc hội thoại khác nhau, sau đó cập nhật Global Network để cải thiện chất lượng câu trả lời dựa trên phản hồi từ người dùng.

1.3 So sánh các thuật toán:

Bảng 1. So sánh các thuật toán Reinforcement Learning

Thuật toán	**Ưu điểm**	**Nhược điểm**	**Ứng dụng trong LLMs**
PPO	- Ổn định, hiệu quả cao. - Phù hợp với không gian hành động lớn.	- Phức tạp, cần mô hình thưởng. - Tốn tài nguyên.	Tinh chỉnh InstructGPT, ChatGPT.

DPO	- Đơn giản, không cần mô hình thưởng. - Hiệu quả với dữ liệu sở thích.	- Phụ thuộc vào dữ liệu sở thích. - Ít linh hoạt hơn PPO.	Tinh chỉnh Llama, Mistral.
TRPO	- Ổn định nhờ trust region. - Phù hợp với không gian hành động lớn.	Phức tạp, tính toán nặng. - Ít phổ biến hơn PPO.	Tinh chỉnh LLMs trong nghiên cứu ban đầu.
A2C	- Hiệu quả nhờ song song hóa.	- Ít ổn định hơn PPO.	Tinh chỉnh hệ thống đối thoại, tạo văn bản.

Phân tích chi tiết:

- Hiệu quả: PPO và DPO vượt trội trong LLMs nhờ khả năng xử lý không gian hành động lớn và tối ưu hóa dựa trên phản hồi người dùng. TRPO và A2C là các phương pháp thay thế, hiệu quả nhưng ít phổ biến hơn.
- Độ phức tạp: DPO đơn giản nhất, trong khi PPO, TRPO, và A2C phức tạp hơn do yêu cầu mô hình thưởng hoặc tính toán ràng buộc.
- Ứng dụng thực tế: PPO và DPO là tiêu chuẩn trong tinh chỉnh LLMs lớn (InstructGPT, Llama). TRPO và A2C phù hợp với nghiên cứu hoặc bài toán cụ thể.

1.4 Kết luận:

Các thuật toán RL dựa trên chính sách (PPO, DPO, TRPO, A2C) đóng vai trò quan trọng trong tinh chỉnh các mô hình ngôn ngữ lớn.

- PPO và DPO là lựa chọn hàng đầu nhờ hiệu quả, ổn định, và khả năng mở rộng trong các bài toán phức tạp.
- TRPO và A2C cung cấp các phương pháp thay thế với ưu điểm về ổn định hoặc song song hóa.

Việc chọn thuật toán hoặc kỹ thuật phụ thuộc vào yêu cầu bài toán, tài nguyên tính toán, và loại dữ liệu có sẵn (phản hồi người dùng hoặc sở thích).

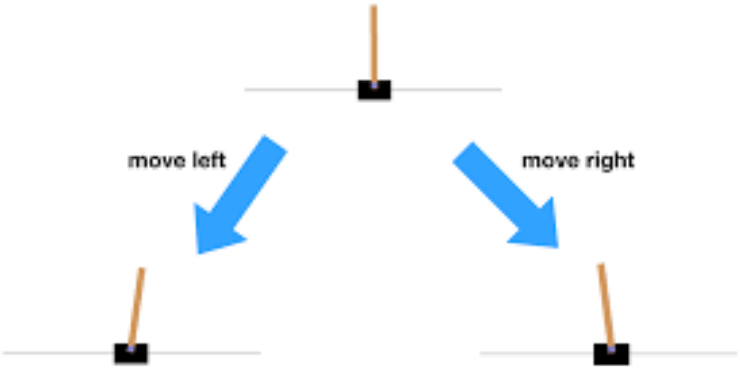
1.5 Thuật toán PPO trong môi trường Cartpole-v1:

Đường link drive chứa mã nguồn:

https://drive.google.com/drive/folders/1lNux2nBx4oqyEOx5c2hWtSMv_GiRzslF?usp=drive_link

1.5.1 Mô hình hóa bài toán:

Bài toán được giải quyết trong tài liệu là huấn luyện một tác tử (agent) sử dụng thuật toán Proximal Policy Optimization (PPO) để chơi trò chơi CartPole-v1 trong môi trường Gym của OpenAI. Mục tiêu là điều khiển một chiếc xe đẩy (cart) có gắn một thanh (pole) sao cho thanh không bị đổ và xe không di chuyển quá xa.



Hình 6. Môi trường Cartpole-v1

Môi trường (CartPole-v1):

- Trạng thái (state): Vector 4 chiều gồm [vị trí xe, vận tốc xe, góc thanh, vận tốc góc thanh].
- Hành động (action): 2 hành động rời rạc (0: đẩy xe sang trái, 1: đẩy xe sang phải).
- Phần thưởng (reward): +1 cho mỗi bước mà thanh không đổ (góc < 15 độ) và xe không ra khỏi giới hạn.
- Điều kiện dừng: Thanh đổ, xe vượt giới hạn, hoặc đạt 500 bước (môi trường được coi là "giải quyết" nếu điểm trung bình trong 25 tập đạt ≥ 500).

Thuật toán PPO: PPO là một thuật toán học tăng cường (Reinforcement Learning) thuộc họ Policy Gradient, tối ưu hóa chính sách (policy) bằng cách cân bằng giữa việc cải thiện hiệu suất và giữ ổn định trong quá trình cập nhật. PPO sử dụng:

- Mạng chính sách (Policy Network): Dự đoán phân phối xác suất hành động từ trạng thái.
- Mạng giá trị (Value Network): Ước lượng giá trị trạng thái (state value) để tính lợi thế (advantage).
- Generalized Advantage Estimation (GAE): Tính lợi thế để ổn định huấn luyện.
- Clipped Surrogate Objective: Giới hạn cập nhật chính sách để tránh thay đổi quá lớn.

Mục tiêu: Tối ưu hóa chính sách để đạt điểm trung bình ≥ 500 trong 25 tập liên tiếp.

1.5.2 Tóm tắt tác dụng các class:

Dưới đây là tóm tắt tác dụng của các class chính trong code:

1.5.2.1 ValueNetwork:

- Tác dụng: Ước lượng giá trị trạng thái (state value) để hỗ trợ tính lợi thế trong PPO. Mạng này dự đoán giá trị kỳ vọng của phần thưởng tích lũy từ một trạng thái.
- Kiến trúc: Mạng nơ-ron với 1 lớp ẩn, sử dụng hàm kích hoạt Sigmoid, đầu ra là một số thực (giá trị trạng thái).
- Cập nhật: Sử dụng hàm mất mát MSE (Mean Squared Error) để tối ưu hóa dự đoán giá trị trạng thái so với phần thưởng chiết khấu.

1.5.2.2 PPOPolicyNetwork:

- Tác dụng: Dự đoán phân phối xác suất hành động từ trạng thái, đại diện cho chính sách của tác tử. Mạng này được cập nhật theo thuật toán PPO để cải thiện chính sách.
- Kiến trúc: Mạng nơ-ron với 3 lớp ẩn, sử dụng ReLU làm hàm kích hoạt, đầu ra là phân phối xác suất qua Softmax.
- Cập nhật: Sử dụng hàm mất mát PPO với kỹ thuật "clipped surrogate objective" để giới hạn cập nhật, đảm bảo ổn định.

1.5.2.3 PPO:

- Tác dụng: Điều phối toàn bộ quá trình huấn luyện, tích hợp mạng giá trị và mạng chính sách, thực hiện các bước thu thập dữ liệu, tính toán lợi thế, và cập nhật mô hình.
- Chức năng chính:
  + Khởi tạo môi trường, mạng giá trị, và mạng chính sách.
  + Chạy các tập (episode) để thu thập dữ liệu (trạng thái, hành động, phần thưởng).
  + Tính toán phần thưởng chiết khấu và lợi thế GAE.
  + Cập nhật mạng giá trị và mạng chính sách.

1.5.3 Phân tích hướng làm:

Hướng tiếp cận của bài toán sử dụng thuật toán PPO để huấn luyện tác tử trong môi trường CartPole-v1. Các bước chính bao gồm:

1.5.3.1 Khởi tạo môi trường và mô hình:

- Tạo môi trường CartPole-v1 với 4 đặc trưng trạng thái và 2 hành động.
- Khởi tạo mạng giá trị (ValueNetwork) và mạng chính sách (PPOPolicyNetwork) với các siêu tham số như tốc độ học, kích thước lớp ẩn, và ngưỡng cắt (epsilon).

1.5.3.2 Thu thập dữ liệu:

- Trong mỗi tập, tác tử tương tác với môi trường bằng cách chọn hành động dựa trên phân phối xác suất từ mạng chính sách.
- Lưu trữ trạng thái, hành động, và phần thưởng cho mỗi bước.

1.5.3.3 Tính toán phần thưởng chiết khấu và lợi thế:

- Sử dụng hàm discount_rewards để tính phần thưởng chiết khấu (discounted rewards).
- Sử dụng hàm calculate_advantages để tính lợi thế GAE, chuẩn hóa để giảm phương sai.

1.5.3.4 Cập nhật mô hình:

- Cập nhật mạng giá trị bằng cách tối ưu hóa MSE giữa giá trị dự đoán và phần thưởng chiết khấu.
- Cập nhật mạng chính sách bằng cách tối ưu hóa hàm mất mát PPO với kỹ thuật cắt tỉ lệ (clipped ratio).

1.5.3.5 Đánh giá và dừng:

- Theo dõi điểm trung bình qua 25 tập.
- Dừng khi điểm trung bình đạt ≥ 500, tức là môi trường được "giải quyết".

CHƯƠNG 2. MACHINE LEARNING TRANSLATION

2.1 Bộ dữ liệu:

- Dữ liệu: Bộ song ngữ tiếng Anh - tiếng Việt (IWSLT15) được công bố tại hội thảo IWSLT 2015 (International Workshop on Spoken Language Translation).
- Định dạng: Cặp câu song ngữ, mỗi dòng gồm một câu tiếng Anh và câu tiếng Việt tương ứng.
- Kích thước: Khoảng 133,000 cặp câu.
- Chủ đề: Nội dung chủ yếu là các bài nói (talks), hội thoại, hoặc bài thuyết trình ngắn.

2.2 Tokenizer và thuật toán BPE:

2.2.1 Tokenizer:

Tokenizer là quá trình chia nhỏ văn bản thành các đơn vị nhỏ hơn gọi là token, là bước tiền xử lý quan trọng trong các tác vụ xử lý ngôn ngữ tự nhiên (NLP) như dịch máy. Token là đơn vị cơ bản mà mô hình học máy (như Transformer) có thể hiểu và xử lý.

Mục đích:

- Chuyển văn bản thô thành dạng số (ID token) để mô hình có thể xử lý
- Giảm kích thước từ vựng (vocabulary) và xử lý các từ hiếm hoặc từ mới (out-of-vocabulary, OOV).
- Chuẩn hóa văn bản để phù hợp với ngôn ngữ cụ thể (ví dụ: xử lý dấu tiếng Việt).

Các loại tokenizer:

- Word-based: Chia văn bản thành các từ dựa trên khoảng trắng hoặc quy tắc ngôn ngữ (ví dụ: "Tôi đi học" → ["Tôi", "đi", "học"]). Nhược điểm: Từ vựng lớn, khó xử lý từ hiếm.

- Character-based: Chia thành từng ký tự (ví dụ: "Tôi" → ["T", "ô", "i"]). Nhược điểm: Chuỗi dài, mất ngữ nghĩa ngữ cảnh.
- Subword-based: Kết hợp giữa từ và ký tự, sử dụng các đơn vị subword (ví dụ: "học sinh" → ["học", "sinh"]). Phổ biến trong các mô hình hiện đại như BERT, Transformer.

Subword tokenization (như BPE, WordPiece, SentencePiece) được ưa chuộng trong dịch máy vì:

- Giảm kích thước từ vựng so với word-based.
- Xử lý từ hiếm bằng cách chia thành các subword nhỏ hơn.
- Phù hợp với các ngôn ngữ không có khoảng trắng rõ ràng (như tiếng Việt, tiếng Trung).

2.2.2 Thuật toán BPE (Byte-Pair Encoding):

2.2.2.1 Khái niệm về BPE:

Byte-Pair Encoding (BPE) là một thuật toán nén dữ liệu, ban đầu được phát triển bởi Philip Gage (1994) để nén văn bản, và sau đó được áp dụng trong NLP bởi Sennrich et al. (2016) cho các tác vụ như dịch máy. Trong NLP, BPE được sử dụng để tạo ra các đơn vị subword (nhỏ hơn từ nhưng lớn hơn ký tự) nhằm biểu diễn văn bản một cách hiệu quả.

Mục đích trong NLP:

- Giảm kích thước từ vựng (vocabulary size) so với phương pháp word-based.
- Xử lý các từ hiếm hoặc từ mới (out-of-vocabulary, OOV) bằng cách chia chúng thành các subword đã biết.
- Phù hợp với các ngôn ngữ có cấu trúc phức tạp hoặc không có khoảng trắng rõ ràng (như tiếng Việt, tiếng Trung).

Ý tưởng triển khai:

- BPE bắt đầu bằng cách chia văn bản thành các ký tự và dần dần gộp các cặp ký tự (hoặc subword) xuất hiện thường xuyên nhất thành các đơn vị lớn hơn.
- Quá trình này lặp lại cho đến khi đạt được kích thước từ vựng mong muốn hoặc số lần gộp tối đa.

2.2.2.2 Nguyên lý hoạt động của BPE:

BPE hoạt động dựa trên việc thống kê tần suất của các cặp ký tự (hoặc subword) và gộp chúng một cách tham lam (greedy). Dưới đây là các bước chi tiết

- Khởi tạo:
  - Chia tất cả các từ trong corpus thành các ký tự riêng lẻ.
  - Thêm một ký hiệu đặc biệt (thường là </w>) vào cuối mỗi từ để đánh dấu ranh giới từ. Điều này giúp BPE phân biệt giữa các ký tự trong từ và giữa các từ.
  - Ví dụ: Từ "low" được biểu diễn thành l o w </w>.
- Thống kê cặp từ vựng:
  - Đếm tần suất xuất hiện của tất cả các cặp ký tự liền kề trong corpus.
  - Một cặp ký tự là hai ký tự (hoặc subword) đứng liền nhau trong một chuỗi.
- Gộp cặp phổ biến nhất:
  - Tìm cặp ký tự có tần suất cao nhất và gộp chúng thành một đơn vị mới (subword).
  - Thay thế tất cả các lần xuất hiện của cặp này trong corpus bằng đơn vị mới.
- Lặp lại:
  - Tiếp tục thống kê cặp ký tự, gộp cặp phổ biến nhất, và cập nhật corpus.
  - Quá trình này lặp lại cho đến khi: Đạt được kích thước từ vựng mong muốn (vocab_size). Hoặc đạt số lần gộp tối đa (số bước được định nghĩa trước).
- Kết quả:

- Tạo ra một từ vựng gồm các ký tự ban đầu và các subword được gộp.
- Từ vựng này được dùng để tokenize văn bản mới bằng cách khớp các subword dài nhất có thể.

2.2.2.3 Ưu điểm của BPE:

- Giảm kích thước từ vựng: Thay vì lưu trữ hàng triệu từ (word-based), BPE tạo từ vựng nhỏ hơn (thường 8k–32k token) bằng cách dùng subword.
- Xử lí từ hiếm và từ mới: Từ không có trong từ vựng được chia thành các subword nhỏ hơn (ví dụ: "unseen" → un + seen).
- Phù hợp với nhiều ngôn ngữ: BPE không phụ thuộc vào khoảng trắng, thích hợp cho tiếng Việt, tiếng Trung, tiếng Đức (có từ ghép dài).
- Hiệu quả tính toán: Subword giúp giảm độ dài chuỗi so với character-based, làm giảm chi phí tính toán trong mô hình Transformer.
- Tính linh hoạt: Có thể điều chỉnh vocab_size để cân bằng giữa biểu diễn ngôn ngữ và hiệu suất.

2.2.2.4 Nhược điểm của BPE:

- Phụ thuộc vào corpus: Nếu corpus không đủ lớn hoặc không đa dạng, BPE có thể tạo ra các subword không có ý nghĩa hoặc không tối ưu.
- Khó tối ưu hóa - Việc chọn vocab_size và số lần gộp là siêu tham số, cần thử nghiệm nhiều lần.
- Mất ngữ nghĩa - Một số subword có thể không mang ý nghĩa đầy đủ (ví dụ: "playing" → play + ing, nhưng ing không có nghĩa độc lập).
- Tốn tài nguyên huấn luyện - Huấn luyện BPE trên corpus lớn có thể tốn thời gian và bộ nhớ.

2.3 BLEU và ROUGE:

2.3.1 BLEU:

2.3.1.1 Tổng quan:

BLEU (Bilingual Evaluation Understudy) là một thước đo tự động phổ biến trong dịch máy và các tác vụ xử lý ngôn ngữ tự nhiên (NLP) để đánh giá chất lượng

của văn bản được sinh ra so với văn bản tham chiếu (reference). Được đề xuất bởi Papineni et al. (2002), BLEU đo lường mức độ tương đồng giữa câu được sinh ra (candidate) và câu tham chiếu dựa trên sự trùng khớp của các n-gram (chuỗi gồm n từ liên tiếp).

2.3.1.2 Nguyên lý hoạt động:

BLEU tính điểm dựa trên hai thành phần chính:

a. Độ chính xác n-gram (n-gram Precision):

- Đếm số lần xuất hiện của các n-gram (thường từ 1-gram đến 4-gram) trong câu được sinh ra có khớp với câu tham chiếu.
- Tính tỷ lệ các n-gram trong câu được sinh ra xuất hiện trong câu tham chiếu.
- Công thức cho độ chính xác n-gram:

$$p_n = \frac{\sum_{n-gram \in candidate} \min\left(Count_{n-gram}^{candidate}, Count_{n-gram}^{reference}\right)}{\sum_{n-gram \in candidate} Count_{n-gram}^{candidate}}$$

Trong đó:

+ $Count_{n-gram}^{candidate}$: Số lần xuất hiện của n-gram trong câu được sinh ra.
+ $Count_{n-gram}^{reference}$: Số lần xuất hiện của n-gram trong câu tham chiếu.
+ min: Lấy giá trị nhỏ hơn để tránh tính trùng lặp quá mức.

b. Phạt độ ngắn (Brevity Penalty, BP):

- Nếu câu được sinh ra ngắn hơn câu tham chiếu, BLEU áp dụng một hình phạt để tránh ưu ái các câu ngắn nhưng thiếu thông tin.
- Công thức:

$$BP = \begin{cases} 1 & ,c > r \\ e^{1-\frac{r}{c}} & ,c \le r \end{cases}$$

Trong đó:

+ c: Độ dài của câu được sinh ra.
+ r: Độ dài của câu tham chiếu.

c. Điểm BLEU tổng quát:

- Kết hợp độ chính xác n-gram (thường từ 1-gram đến 4-gram) bằng trung bình hình học, sau đó nhân với BP:

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} \mathrm{w}_n \log p_n\right)$$

Trong đó:

+ w_n : Trọng số của n-gram, thường là $\frac{1}{N}$ (ví dụ: $\frac{1}{4}$ cho 1-gram đến 4-gram).
+ N: Độ dài n-gram tối đa (thường là 4).

2.3.1.3 Ưu và nhược điểm:

a. Ưu điểm:

- Đơn giản và nhanh chóng: BLEU dễ triển khai và tính toán, phù hợp cho đánh giá tự động trên tập dữ liệu lớn.
- Phổ biến: Được sử dụng rộng rãi trong dịch máy, cho phép so sánh giữa các mô hình.
- Hiệu quả với n-gram: Đo lường tốt sự tương đồng về từ vựng và cấu trúc câu ngắn.

b. Nhược điểm:

- Không đánh giá ngữ nghĩa: BLEU chỉ dựa trên sự trùng khớp n-gram, không xem xét ý nghĩa hoặc ngữ cảnh của câu.
- Phụ thuộc vào câu tham chiếu: Điểm BLEU có thể thay đổi tùy thuộc vào số lượng và chất lượng câu tham chiếu.
- Không xử lý tốt sự đa dạng: BLEU không đánh giá tốt các câu có từ đồng nghĩa hoặc cách diễn đạt khác nhưng vẫn đúng.
- Ưu ái câu ngắn: Mặc dù có BP, BLEU vẫn có thể ưu ái các câu ngắn hơn so với câu dài nhưng chất lượng cao.

2.3.1.4 Ứng dụng:

- Đánh giá chất lượng dịch máy (machine translation) như trong bộ dữ liệu IWSLT15 (tiếng Anh - tiếng Việt).
- So sánh hiệu suất giữa các mô hình dịch máy (ví dụ: Transformer, LSTM).
- Đánh giá các tác vụ sinh văn bản khác như tóm tắt văn bản hoặc đối thoại.

2.3.2 ROUGE:

2.3.2.1 Tổng quan:

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) là một tập hợp các thước đo được đề xuất bởi Lin (2004) để đánh giá chất lượng văn bản được sinh ra, đặc biệt trong các tác vụ như tóm tắt văn bản (text summarization) và dịch máy. ROUGE tập trung vào việc đo lường sự trùng khớp giữa văn bản được sinh ra và văn bản tham chiếu dựa trên recall (độ bao phủ).

2.3.2.2 Các biến thể chính của ROUGE:

a. <u>ROUGE-N:</u>

- Đo lường sự trùng khớp của n-gram (thường 1-gram, 2-gram) giữa văn bản được sinh ra và văn bản tham chiếu.
- Công thức:

$$ROUGE - N = \frac{\sum_{n-gram \in reference} \min\left(Count_{n-gram}^{candidate}, Count_{n-gram}^{reference}\right)}{\sum_{n-gram \in reference} Count_{n-gram}^{reference}}$$

Trong đó:

+ $Count_{n-gram}^{candidate}$: Số lần xuất hiện của n-gram trong câu được sinh ra.

+ $Count_{n-gram}^{reference}$: Số lần xuất hiện của n-gram trong câu tham chiếu.

b. <u>ROUGE-L:</u>

- Dựa trên chuỗi con chung dài nhất (Longest Common Subsequence, LCS) giữa văn bản được sinh ra và văn bản tham chiếu.

- ROUGE-L tập trung vào cấu trúc và trật tự từ, không yêu cầu các từ phải liên tiếp.
- Công thức:

$$ROUGE - L = \frac{\left(1+\beta^2\right)\cdot R_{LCS}\cdot P_{LCS}}{R_{LCS}+\beta^2\cdot P_{LCS}}$$

Trong đó:

+ $R_{LCS} = \frac{LCS(X,Y)}{m}$: Recall của LCS, với m là độ dài văn bản tham chiếu:

+ $P_{LCS} = \frac{LCS(X,Y)}{n}$: Precision của LCS, với n là độ dài văn bản được sinh ra.

+ $LCS(X,Y)$: Độ dài chuỗi con chung dài nhất giữa X (văn bản tham chiếu) và Y (văn bản được sinh ra).

+ β: Hệ số cân bằng giữa recall và precision (thường $\beta = 1$).

c. <u>ROUGE-S:</u>

- Đo lường sự trùng khớp của các cặp từ không liên tiếp (skip-bigram) trong văn bản.
- Phù hợp để đánh giá các văn bản có trật tự từ linh hoạt hơn.

2.3.2.3 Ưu và nhược điểm:

a. <u>Ưu điểm:</u>

- Tập trung vào recall: ROUGE đánh giá tốt mức độ bao phủ thông tin từ văn bản tham chiếu, phù hợp với tóm tắt văn bản.
- Linh hoạt: ROUGE-L và ROUGE-S cho phép đánh giá cấu trúc và trật tự từ, không chỉ n-gram cố định.
- Hỗ trợ đa dạng tham chiếu: ROUGE hoạt động tốt với nhiều câu tham chiếu, phản ánh tính đa dạng của ngôn ngữ.

b. Nhược điểm:

- Không đánh giá ngữ nghĩa: Tương tự BLEU, ROUGE chủ yếu dựa trên sự trùng khớp từ vựng, không xem xét ý nghĩa.
- Phụ thuộc vào tham chiếu: Chất lượng đánh giá phụ thuộc vào văn bản tham chiếu, có thể không phản ánh đầy đủ tính đúng đắn của văn bản được sinh ra.
- Hạn chế với văn bản sáng tạo: ROUGE không đánh giá tốt các văn bản có cách diễn đạt khác biệt nhưng vẫn đúng về nội dung.

2.3.2.4 Ứng dụng:

- Đánh giá chất lượng tóm tắt văn bản (summarization), đặc biệt trong các hệ thống tự động.
- Đánh giá dịch máy, đặc biệt khi cần đo lường mức độ bao phủ thông tin.
- So sánh hiệu suất các mô hình NLP trong các tác vụ sinh văn bản.

2.3.3 So sánh BLEU và ROUGE:

Bảng 2. So sánh BLEU và ROUGE

Thước đo	Loại	Ưu điểm	Nhược điểm	Ứng dụng chính
BLEU	Precision-based	- Đơn giản, nhanh. - Hiệu quả với dịch máy nhờ đo lường sự trùng khớp n-gram chính xác.	- Không đánh giá ngữ nghĩa. - Ưu ái câu ngắn. - Không xử lý tốt cách diễn đạt khác nhau.	Dịch máy, so sánh mô hình dịch.
ROUGE-N	Recall-based	- Đo lường tốt mức độ bao phủ n-gram. - Phù hợp với tóm tắt văn bản	- Chỉ dựa trên n-gram, bỏ qua cấu trúc dài hơn. - Không đánh giá ngữ nghĩa.	Tóm tắt văn bản, đánh giá bao phủ từ vựng.

		nhờ tập trung vào recall.		
ROUGE-L	Recall-based (LCS)	- Linh hoạt, đánh giá cấu trúc và trật tự từ thông qua LCS. - Phù hợp với văn bản có trật tự từ linh hoạt.	- Phụ thuộc vào tham chiếu. - Không đánh giá ngữ nghĩa. - Phức tạp hơn ROUGE-N.	Tóm tắt văn bản, dịch máy với cấu trúc linh hoạt.

2.3.4 Phân tích:

BLEU tập trung vào độ chính xác (precision) của n-gram, phù hợp cho dịch máy nơi sự trùng khớp từ vựng và cấu trúc câu ngắn là quan trọng. Tuy nhiên, nó có thể bỏ qua các câu đúng về ý nghĩa nhưng sử dụng từ đồng nghĩa hoặc cách diễn đạt khác, khiến nó kém linh hoạt trong các tác vụ yêu cầu sự đa dạng.

ROUGE-N nhấn mạnh recall, đo lường mức độ bao phủ các n-gram từ văn bản tham chiếu. Điều này làm cho ROUGE-N hiệu quả trong tóm tắt văn bản, nơi việc giữ lại thông tin quan trọng là ưu tiên, nhưng nó bỏ qua cấu trúc câu dài hơn hoặc trật tự từ không liên tiếp.

ROUGE-L sử dụng chuỗi con chung dài nhất (LCS) để đánh giá cấu trúc và trật tự từ, không yêu cầu các từ phải liên tiếp. Điều này làm cho ROUGE-L linh hoạt hơn ROUGE-N, đặc biệt trong các tác vụ như tóm tắt văn bản hoặc dịch máy, nơi cách diễn đạt có thể thay đổi nhưng vẫn giữ được ý nghĩa. Tuy nhiên, ROUGE-L phức tạp hơn và vẫn không giải quyết được vấn đề ngữ nghĩa.

Trong bối cảnh bộ dữ liệu IWSLT15 (tiếng Anh - tiếng Việt), BLEU là thước đo chính để đánh giá dịch máy do khả năng đo lường sự tương đồng từ vựng chính xác. ROUGE-N có thể được sử dụng bổ sung để đánh giá mức độ bao phủ thông tin, đặc biệt khi các câu dịch có sự khác biệt nhỏ về từ vựng. ROUGE-L phù hợp khi cần

đánh giá cấu trúc câu linh hoạt, chẳng hạn khi các câu dịch giữ nguyên ý nghĩa nhưng thay đổi trật tự từ.

2.3.5 Kết luận:

BLEU, ROUGE-N và ROUGE-L là các thước đo quan trọng trong đánh giá các tác vụ NLP, đặc biệt là dịch máy và tóm tắt văn bản. BLEU ưu tiên độ chính xác n-gram và phù hợp với dịch máy, trong khi ROUGE-N và ROUGE-L tập trung vào recall, với ROUGE-L linh hoạt hơn nhờ sử dụng LCS.

Trong các dự án như dịch máy tiếng Anh - tiếng Việt, việc kết hợp cả ba thước đo có thể cung cấp cái nhìn toàn diện hơn về hiệu suất mô hình, với BLEU đánh giá độ chính xác, ROUGE-N đo lường bao phủ từ vựng, và ROUGE-L xem xét cấu trúc câu.

2.4 Triển khai các mô hình dịch máy:

2.4.1 Mô hình Encoder-Decoder Transformer No-Pretrained:

Dưới đây là các bước và phương pháp thực hiện mô hình Transformer cho nhiệm vụ dịch máy tiếng Anh - tiếng Việt, dựa trên bộ dữ liệu. Quy trình bao gồm chuẩn bị môi trường, xử lý dữ liệu, huấn luyện, đánh giá và thử nghiệm dịch thuật.

2.4.1.1 Cài đặt môi trường:

Mô tả:

- Cài đặt các thư viện cần thiết để xử lý dữ liệu, huấn luyện mô hình và đánh giá hiệu suất.

Phương pháp:

- Sử dụng pip để cài đặt các thư viện:
  + datasets: Để tải và quản lý bộ dữ liệu.
  + rouge-score và nltk: Để tính điểm ROUGE và BLEU.
  + Các thư viện khác (torch, numpy, sklearn, tqdm, sentencepiece, v.v.) được sử dụng để huấn luyện và xử lý dữ liệu.

2.4.1.2 Tải và chuẩn bị dữ liệu:

Mô tả:

- Tải bộ dữ liệu song ngữ tiếng Anh - tiếng Việt từ tệp văn bản và thực hiện tiền xử lý.

Phương pháp:

- Đọc dữ liệu: Đọc các tệp train.en.txt (tiếng Anh) và train.vi.txt (tiếng Việt) từ bộ dữ liệu IWSLT15.
- Tiền xử lý:
  + Chuẩn hóa Unicode (NFC) để đảm bảo tính nhất quán của ký tự tiếng Việt.
  + Loại bỏ khoảng trắng thừa và chuẩn hóa khoảng trắng bằng re.sub.
  + Lọc các câu có độ dài từ 5 đến 128 từ để đảm bảo chất lượng dữ liệu.
- Kết quả: Sau khi lọc, còn lại 129,787 cặp câu song ngữ.

2.4.1.3 Tạo Tokenizer với SentencePiece:

Mô tả:

- Tạo tokenizer BPE (Byte-Pair Encoding) để mã hóa dữ liệu văn bản tiếng Anh và tiếng Việt.

Phương pháp:

- Tạo corpus: Ghi toàn bộ dữ liệu tiếng Anh và tiếng Việt vào một tệp corpus.txt.
- Huấn luyện tokenizer:
  + Sử dụng sentencepiece để huấn luyện mô hình BPE với kích thước từ vựng 8,000.
  + Cấu hình: model_type=BPE, character_coverage=1, max_sentence_length=4192.
- Kết quả: Tạo tệp bpe_model.model và bpe_model.vocab để mã hóa và giải mã văn bản.

2.4.1.4 Tiền xử lý và chia dữ liệu:

Mô tả:

- Mã hóa dữ liệu văn bản và chia thành các tập huấn luyện, kiểm tra và đánh giá.

Phương pháp:

- Tokenizer: Sử dụng SentencePieceProcessor để mã hóa các câu thành subword tokens.
- Tạo từ điển: Xây dựng từ điển word2id và id2word cho cả tiếng Anh (inp_lang) và tiếng Việt (tar_lang), bao gồm các token đặc biệt (<SOS>, <EOS>, <PAD>, <UNK>).
- Chuyển đổi dữ liệu: Chuyển các câu thành vector số (dựa trên từ điển) với độ dài tối đa cố định (max_len).
- Chia dữ liệu: Sử dụng train_test_split từ sklearn để chia dữ liệu thành:
  + Tập huấn luyện: ~80% (103,829 mẫu).
  + Tập kiểm tra: ~10% (12,979 mẫu).
  + Tập đánh giá: ~10% (12,979 mẫu).

2.4.1.5 Chuẩn bị DataLoader:

Mô tả:

- Tạo DataLoader để cung cấp dữ liệu theo batch trong quá trình huấn luyện và kiểm tra.

Phương pháp:

- Batch size: Đặt kích thước batch là 64 để cân bằng giữa hiệu suất và bộ nhớ GPU.
- Shuffle: Bật chế độ xáo trộn cho tập huấn luyện để cải thiện tính tổng quát.
- DataLoader: Tạo DataLoader cho các tập huấn luyện, kiểm tra và đánh giá.

2.4.1.6 Khởi tạo mô hình Transformer:

Mô tả:

- Xây dựng mô hình Transformer cho dịch máy.

Phương pháp:

- Kiến trúc:
  + Sử dụng torch.nn.Transformer với các tham số:
    - d_model=512: Kích thước embedding.
    - nhead=8: Số đầu attention.
    - num_encoder_layers=6, num_decoder_layers=6: Số lớp mã hóa và giải mã.
    - dim_feedforward=2048: Kích thước tầng feed-forward.
  + Thêm tầng embedding và positional encoding.
- Mask: Sử dụng generate_square_subsequent_mask để ngăn decoder nhìn vào các token tương lai.
- Device: Chuyển mô hình sang GPU (nếu có) để tăng tốc độ xử lý.

2.4.1.7 Huấn luyện mô hình:

Mô tả:

- Huấn luyện mô hình Transformer trên dữ liệu IWSLT15.

Phương pháp:

- Loss: Sử dụng CrossEntropyLoss với ignore_index=0 để bỏ qua token <PAD>.
- Optimizer: Sử dụng Adam với tốc độ học (lr) là 0.0001.
- Số epoch: Không được chỉ định rõ trong mã, giả định khoảng 10-20 epoch.
- Lưu mô hình: Lưu mô hình sau khi huấn luyện (không có cơ chế dừng sớm trong mã).

2.4.1.8 Đánh giá mô hình:

Mô tả:

- Đánh giá mô hình trên tập kiểm tra bằng các chỉ số ROUGE và BLEU.

Phương pháp:

- Dịch văn bản: Sử dụng mô hình để sinh câu tiếng Việt từ câu tiếng Anh theo từng token, dừng khi gặp <EOS>.
- Chỉ số đánh giá:

  + ROUGE: Tính bằng rouge_scorer cho ROUGE-1 và ROUGE-L trên 200 mẫu.

    - Kết quả: ROUGE-1 ~0.6367, ROUGE-L ~0.5116.

  BLEU: Tính bằng sentence_bleu với smoothing (method4) trên 12,979 mẫu.

    - Kết quả: BLEU ~17.8661.
- Giải mã BPE: Chuyển subword tokens thành văn bản bằng decode_bpe, thay thế ▁ bằng khoảng trắng.

2.4.1.9 Thử nghiệm dịch thuật:

Mô tả:

- Thử nghiệm khả năng dịch của mô hình trên tập kiểm tra.

Phương pháp:

- Dịch trên tập kiểm tra: Dịch 20 câu đầu tiên từ tập kiểm tra, hiển thị câu đầu vào (tiếng Anh) và bản dịch (tiếng Việt).
- Giải mã BPE: Sử dụng decode_bpe để chuyển subword tokens thành văn bản hoàn chỉnh.
- Kết quả: In câu đầu vào và bản dịch, ví dụ:

  + Input: "And because of those four characteristics they are able to filter it as it comes into the country."

  + Translation: "Và bởi vì 4 loại độc tố đó có thể phân biệt được nó như là đất nước."

2.4.2 Mô hình Encoder-Decoder Transformer Pretrained:

Dưới đây là các bước và phương pháp thực hiện mô hình dịch máy Helsinki-NLP MarianMT (dựa trên bộ dữ liệu IWSLT15 tiếng Anh - tiếng Việt). Quy trình bao gồm chuẩn bị dữ liệu, tiền xử lý, huấn luyện, đánh giá và trực quan hóa kết quả.

2.4.2.1 Cài đặt môi trường:

Mô tả:

- Cài đặt các thư viện cần thiết để xử lý dữ liệu, huấn luyện mô hình và đánh giá hiệu suất.

Phương pháp:

- Sử dụng pip để cài đặt các thư viện:
  + sacrebleu: Thư viện tính điểm BLEU.
  + rouge-score: Thư viện tính điểm ROUGE.
  + Các thư viện khác (transformers, datasets, pandas, torch, v.v.) được sử dụng để xử lý dữ liệu và huấn luyện mô hình.

2.4.2.2 Tải và chia dữ liệu:

Mô tả:

- Tải bộ dữ liệu song ngữ tiếng Anh - tiếng Việt từ tệp văn bản và chia thành tập huấn luyện, kiểm tra và đánh giá.

Phương pháp:

- Đọc dữ liệu: Đọc các tệp train.en.txt (tiếng Anh) và train.vi.txt (tiếng Việt) từ bộ dữ liệu IWSLT15.
- Làm sạch dữ liệu: Sử dụng html.unescape để giải mã các thực thể HTML và strip để loại bỏ khoảng trắng thừa.
- Kiểm tra tính toàn vẹn: Đảm bảo số dòng tiếng Anh và tiếng Việt khớp nhau.
- Chia dữ liệu: Sử dụng train_test_split từ sklearn để chia dữ liệu thành:
  + Tập huấn luyện (80%): 106,653 mẫu.
  + Tập kiểm tra (10%): 13,332 mẫu.
  + Tập đánh giá (10%): 13,332 mẫu.

2.4.2.3 Tiền xử lý dữ liệu:

Mô tả:

- Chuyển đổi dữ liệu văn bản thô thành định dạng mà mô hình MarianMT có thể xử lý.

Phương pháp:

- Tokenizer: Sử dụng MarianTokenizer từ Helsinki-NLP/opus-mt-en-vi để mã hóa văn bản tiếng Anh (nguồn) và tiếng Việt (mục tiêu).
- Xử lý batch: Áp dụng mã hóa với độ dài tối đa 128, cắt bớt (truncation) và đệm (padding) để đảm bảo các câu có cùng độ dài.
- Định dạng đầu ra: Tạo các trường input_ids, attention_mask (cho câu nguồn) và labels (cho câu mục tiêu).
- Dataset: Chuyển DataFrame thành Dataset của thư viện datasets để xử lý hiệu quả.

2.4.2.4 Chuẩn bị DataLoader:

Mô tả:

- Tạo DataLoader để cung cấp dữ liệu theo batch trong quá trình huấn luyện và kiểm tra.

Phương pháp:

- Collate function: Chuyển đổi các mẫu dữ liệu thành tensor (input_ids, attention_mask, labels) để phù hợp với định dạng đầu vào của mô hình.
- Batch size: Đặt kích thước batch là 16 để cân bằng giữa hiệu suất và bộ nhớ GPU.
- Shuffle: Bật chế độ xáo trộn cho tập huấn luyện để cải thiện tính tổng quát.

2.4.2.5 Khởi tạo mô hình và Tokenizer:

Mô tả:

- Tải mô hình và tokenizer từ kho lưu trữ của Helsinki-NLP.

Phương pháp:

- Mô hình: Sử dụng MarianMTModel từ Helsinki-NLP/opus-mt-en-vi, một mô hình Transformer được huấn luyện trước cho dịch máy tiếng Anh - tiếng Việt.
- Tokenizer: Tải MarianTokenizer tương ứng để mã hóa và giải mã văn bản.
- Device: Chuyển mô hình sang GPU (nếu có) để tăng tốc độ xử lý.

2.4.2.6 Huấn luyện mô hình:

Mô tả:

- Tinh chỉnh mô hình trên dữ liệu IWSLT15 với cơ chế dừng sớm và mixed precision để tối ưu hóa hiệu suất.

Phương pháp:

- Optimizer: Sử dụng AdamW với tốc độ học (lr) là 2×10^{-5}.
- Mixed Precision: Sử dụng torch.cuda.amp (GradScaler và autocast) để giảm thời gian huấn luyện và sử dụng bộ nhớ hiệu quả.
- Early Stopping: Dừng huấn luyện nếu loss trên tập kiểm tra không cải thiện sau 2 epoch (patience=2).
- Lưu mô hình: Lưu mô hình tốt nhất dựa trên loss kiểm tra thấp nhất.
- Số epoch: Tối đa 10 epoch, nhưng có thể dừng sớm.

2.4.2.7 Đánh giá mô hình:

Mô tả:

- Đánh giá mô hình trên tập kiểm tra bằng các chỉ số BLEU, ROUGE-1 và ROUGE-L.

Phương pháp:

- Dịch văn bản: Sử dụng hàm translate để dịch từng câu tiếng Anh sang tiếng Việt.
- Chỉ số đánh giá:
  + BLEU: Tính bằng sacrebleu.corpus_bleu để đo độ chính xác n-gram.

+ ROUGE-1 và ROUGE-L: Tính bằng rouge_scorer.RougeScorer để đo độ bao phủ và cấu trúc câu.

- Số mẫu: Đánh giá trên một số mẫu (ví dụ: 100 mẫu) để giảm thời gian xử lý.
- Kết quả: In điểm BLEU, ROUGE-1 F1 và ROUGE-L F1 trung bình.

2.4.2.8 Trực quan hóa kết quả:

Mô tả:

- Vẽ biểu đồ loss huấn luyện và kiểm tra để phân tích hiệu suất mô hình.

Phương pháp:

- Biểu đồ: Sử dụng matplotlib.pyplot để vẽ biểu đồ đường thể hiện loss huấn luyện và kiểm tra qua các epoch.
- Lưu kết quả: Lưu biểu đồ dưới dạng loss_plot.png.

2.4.2.9 Lưu và tải mô hình:

Mô tả:

- Lưu mô hình tốt nhất trong quá trình huấn luyện và tải lại để sử dụng.

Phương pháp:

- Lưu mô hình: Sử dụng save_pretrained để lưu mô hình và tokenizer vào thư mục ./opus-en-vi-best.
- Tải mô hình: Sử dụng from_pretrained để tải mô hình và tokenizer từ thư mục đã lưu.
- Chuyển sang chế độ đánh giá: Đặt model.eval() để sử dụng mô hình trong chế độ suy luận.

2.4.3 Mô hình GPT Pretrained:

Dưới đây là các bước và phương pháp thực hiện mô hình GPT-2 cho nhiệm vụ dịch máy tiếng Anh - tiếng Việt, dựa trên bộ dữ liệu IWSLT15. Quy trình bao gồm chuẩn bị môi trường, xử lý dữ liệu, huấn luyện, đánh giá và thử nghiệm dịch thuật.

2.4.3.1 Cài đặt môi trường:

Mô tả:

- Cài đặt các thư viện cần thiết để xử lý dữ liệu, huấn luyện mô hình và đánh giá hiệu suất.

Phương pháp:

- Sử dụng pip để cài đặt các thư viện:
  + evaluate: Để tải các chỉ số đánh giá như ROUGE và BLEU.
  + sacrebleu và rouge_score: Để tính điểm BLEU và ROUGE.
  + Các thư viện khác (transformers, torch, sklearn, tqdm, v.v.) được sử dụng để huấn luyện và xử lý dữ liệu.

2.4.3.2 Tải và chia dữ liệu:

Mô tả:

- Tải bộ dữ liệu song ngữ tiếng Anh - tiếng Việt từ tệp văn bản và chia thành các tập huấn luyện, kiểm tra và đánh giá.

Phương pháp:

- Đọc dữ liệu: Đọc các tệp train.en.txt (tiếng Anh) và train.vi.txt (tiếng Việt) từ bộ dữ liệu IWSLT15.
- Làm sạch dữ liệu: Sử dụng html.unescape để giải mã các thực thể HTML và strip để loại bỏ khoảng trắng thừa.
- Định dạng dữ liệu: Kết hợp câu tiếng Anh và tiếng Việt thành một chuỗi với định dạng [EN] câu_tiếng_Anh [VI] câu_tiếng_Việt.
- Chia dữ liệu: Sử dụng train_test_split từ sklearn để chia dữ liệu thành:
  + Tập huấn luyện (80%): 106,653 mẫu.
  + Tập kiểm tra (10%): 13,332 mẫu.
  + Tập đánh giá (10%): 13,332 mẫu.

2.4.3.3 Tiền xử lý dữ liệu:

Mô tả:

- Chuẩn bị dữ liệu văn bản để phù hợp với mô hình GPT-2 bằng cách mã hóa và tạo dataset tùy chỉnh.

Phương pháp:

- Tokenizer: Sử dụng GPT2Tokenizer từ mô hình gpt2 được huấn luyện trước.
  + Gán pad_token bằng eos_token để xử lý đệm.
  + Thêm các token đặc biệt [EN] và [VI] để đánh dấu câu nguồn và mục tiêu.
- Mã hóa dữ liệu: Mã hóa dữ liệu với độ dài tối đa 128, sử dụng đệm (padding) và cắt bớt (truncation).
- Dataset tùy chỉnh: Tạo lớp TranslationDataset kế thừa từ torch.utils.data.Dataset, trả về input_ids, attention_mask và labels (sao chép từ input_ids).

2.4.3.4 Chuẩn bị DataLoader:

Mô tả:

- Tạo DataLoader để cung cấp dữ liệu theo batch trong quá trình huấn luyện và kiểm tra.

Phương pháp:

- Batch size: Đặt kích thước batch là 8 để cân bằng giữa hiệu suất và bộ nhớ GPU.
- Shuffle: Bật chế độ xáo trộn cho tập huấn luyện để cải thiện tính tổng quát.
- DataLoader: Tạo DataLoader cho các tập huấn luyện, kiểm tra và đánh giá.

2.4.3.5 Khởi tạo mô hình:

Mô tả:

- Tải mô hình GPT-2 và điều chỉnh để phù hợp với tokenizer đã tùy chỉnh.

Phương pháp:

- Mô hình: Sử dụng GPT2LMHeadModel từ gpt2, một mô hình ngôn ngữ dựa trên Transformer.

- Điều chỉnh embedding: Gọi resize_token_embeddings để cập nhật kích thước embedding tương ứng với tokenizer đã thêm token đặc biệt.
- Device: Chuyển mô hình sang GPU (nếu có) để tăng tốc độ xử lý.

2.4.3.6 Huấn luyện mô hình:

Mô tả:

- Tinh chỉnh mô hình GPT-2 trên dữ liệu IWSLT15 với cơ chế dừng sớm, mixed precision và lịch trình học tập.

Phương pháp:

- Optimizer: Sử dụng AdamW với tốc độ học (lr) là 2×10^{-5}.
- Scheduler: Sử dụng ReduceLROnPlateau để giảm tốc độ học khi loss kiểm tra không cải thiện (factor=0.5, patience=1).
- Mixed Precision: Sử dụng torch.cuda.amp (GradScaler và autocast) để giảm thời gian huấn luyện và tối ưu hóa bộ nhớ.
- Early Stopping: Dừng huấn luyện nếu loss kiểm tra không cải thiện sau 3 epoch (patience=3).
- Số epoch: Tối đa 15 epoch, nhưng có thể dừng sớm.
- Lưu mô hình: Lưu mô hình tốt nhất (dựa trên loss kiểm tra thấp nhất) và mô hình cuối cùng.

2.4.3.7 Đánh giá mô hình:

Mô tả:

- Đánh giá mô hình trên tập kiểm tra bằng các chỉ số ROUGE và BLEU.

Phương pháp:

- Dịch văn bản: Sử dụng mô hình để sinh câu tiếng Việt từ câu tiếng Anh với cấu trúc [EN] câu_tiếng_Anh [VI].
- Chỉ số đánh giá:
  + ROUGE: Tính bằng evaluate.load('rouge') để đo độ bao phủ n-gram và cấu trúc câu.

+ BLEU: Tính bằng evaluate.load('sacrebleu') để đo độ chính xác n-gram.

- Số mẫu: Đánh giá trên 200 hoặc 2000 mẫu để cân bằng giữa thời gian và độ chính xác.
- Kết quả:
  + Trên 200 mẫu: ROUGE-1 ~0.55, ROUGE-2 ~0.36, ROUGE-L ~0.48, BLEU ~13.40.
  + (Kết quả trên 2000 mẫu không được cung cấp trong mã).

2.4.3.8 Thử nghiệm dịch thuật:

Mô tả:

- Thử nghiệm khả năng dịch của mô hình trên tập kiểm tra hoặc câu tùy ý.

Phương pháp:

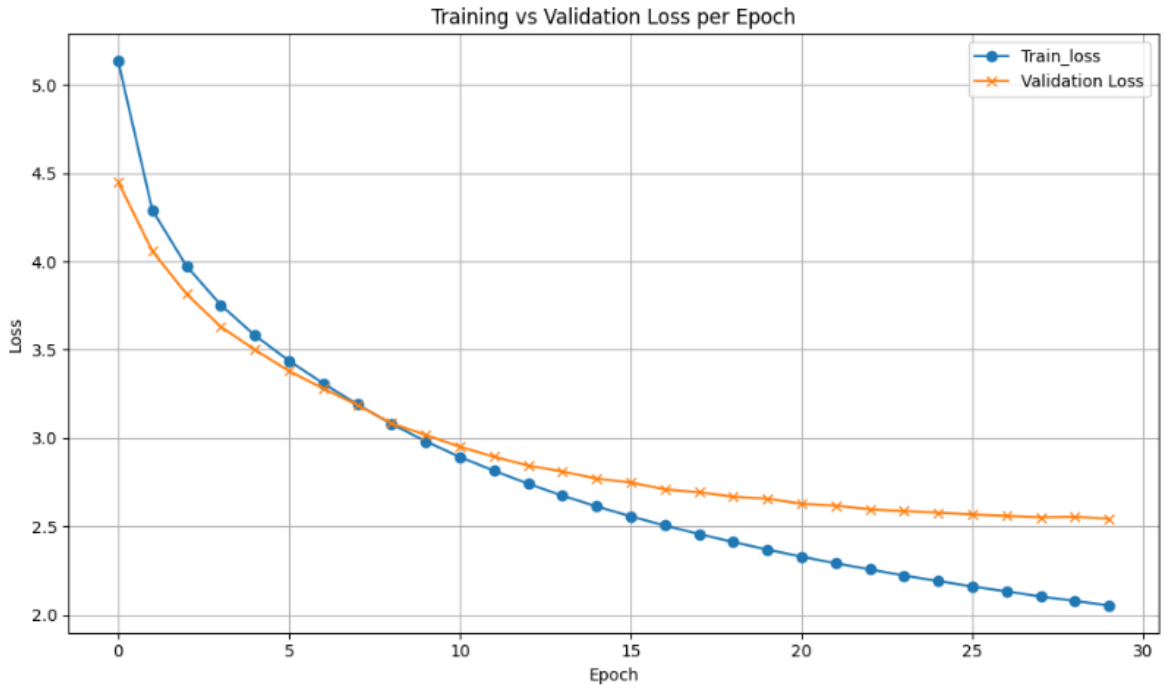
- Dịch trên tập kiểm tra: Dịch 5 câu đầu tiên từ tập kiểm tra, so sánh bản dịch với câu tham chiếu.
- Dịch câu tùy ý: Dịch một câu tiếng Anh do người dùng cung cấp (ví dụ: "Hello, how").
- Sinh văn bản: Sử dụng model.generate với num_beams=5 và early_stopping=True để tạo bản dịch chất lượng cao.
- Kết quả: In câu đầu vào, bản dịch và câu tham chiếu (nếu có).

CHƯƠNG 3. NHẬN ĐỊNH VÀ ĐÁNH GIÁ MÔ HÌNH

Đường link drive chứa mã nguồn:

https://drive.google.com/drive/folders/1lNux2nBx4oqyEOx5c2hWtSMv_GiRzslF?usp=drive_link

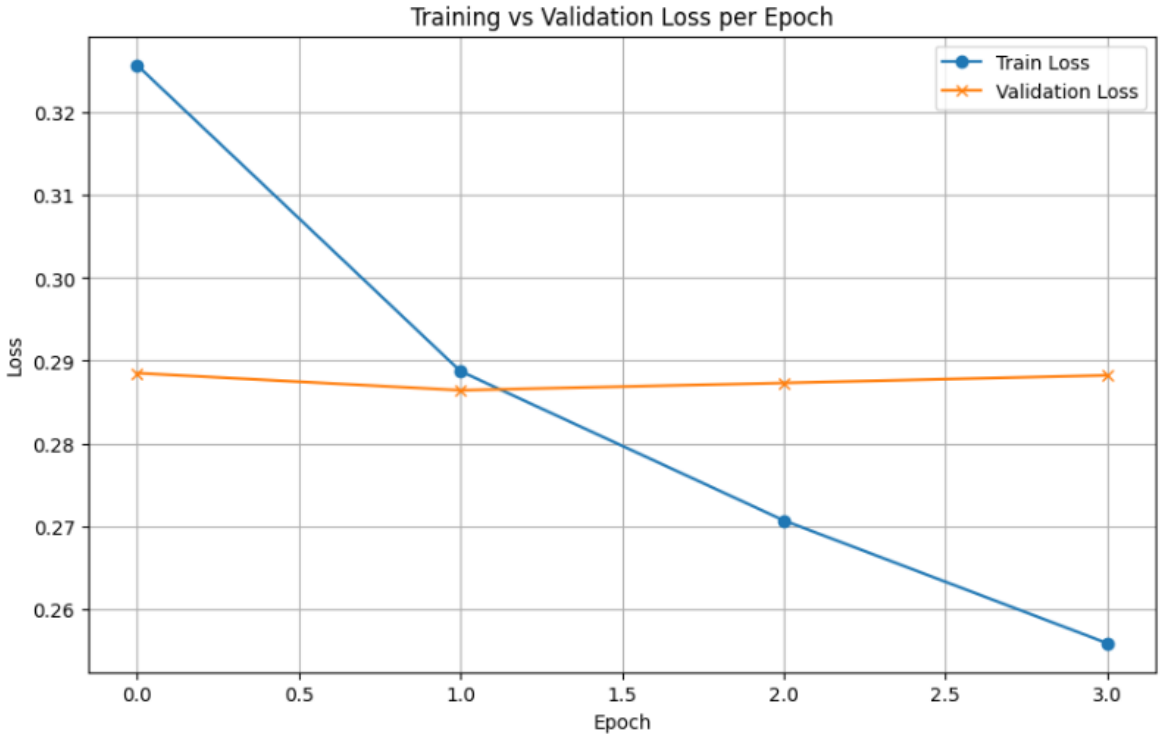
3.1 Biểu đồ huấn luyện:



Hình 7. Biểu đồ huấn luyện Transformer từ đầu

- Mô hình hội tụ tốt:
  + Cả đường Train loss (xanh dương) và Validation loss (cam) đều giảm đều trong 10 epoch đầu.
  + Điều này cho thấy mô hình học được quy luật từ dữ liệu huấn luyện một cách hiệu quả.
- Không bị overfitting nghiêm trọng:
  + Mặc dù sau khoảng epoch 10, validation loss bắt đầu giảm chậm hơn so với train loss, nhưng hai đường vẫn khá gần nhau → chưa có dấu hiệu rõ ràng của overfitting.
  + Mô hình vẫn tổng quát hóa tốt trên dữ liệu chưa thấy (validation).

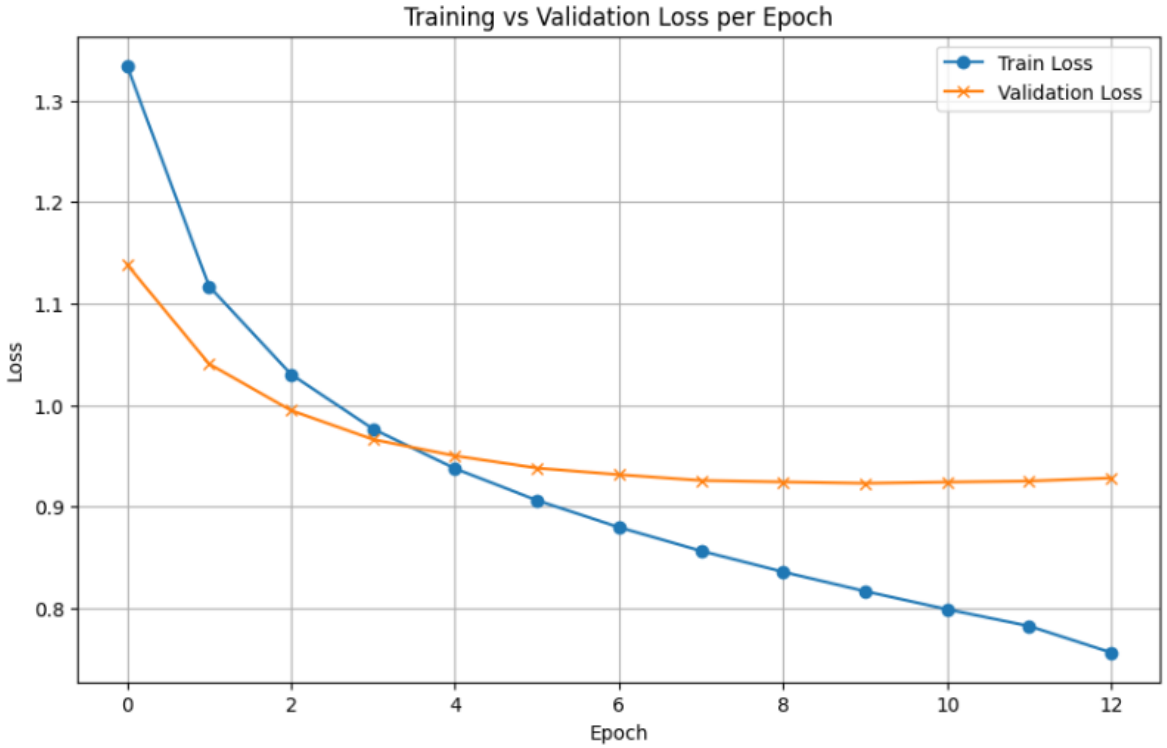
- Hiệu quả cải thiện giảm dần:
  + Sau epoch ~20, cả hai đường bắt đầu tiệm cận, cho thấy mô hình gần đạt giới hạn học được từ dữ liệu hiện tại.
  + Nếu muốn cải thiện thêm, cần:
    - Dữ liệu phong phú hơn.
    - Kiến trúc tốt hơn (tăng layer, attention head).
    - Hoặc fine-tune từ mô hình pretrained.



Hình 8. Biểu đồ huấn luyện Transformer có pretrained

- Train loss giảm đều → fine-tuning hoạt động: Đường Train Loss (xanh) giảm rõ theo từng epoch → mô hình đang tinh chỉnh tốt từ pretrained weights.
- Validation loss gần như giữ nguyên: Đường Validation Loss (cam) dao động nhẹ quanh mức 0.288–0.289, không giảm theo Train Loss → mô hình chưa tối ưu hóa tốt cho tập kiểm định.

- Không có overfitting rõ rệt, nhưng cũng không có generalization tốt: Train loss giảm, val loss đi ngang → mô hình đang nhớ lại kiến thức đã học, nhưng chưa học thêm gì đáng kể từ dữ liệu mới.
- Tuy nhiên, vì mô hình đã được huấn luyện sẵn trên dữ liệu có cấu trúc tương đồng, validation loss có thể đạt mức thấp ngay từ đầu và không giảm nhiều thêm. Tuy nhiên, BLEU/ROUGE vẫn cho thấy kết quả hợp lý.



Hình 9. Biểu đồ huấn luyện GPT có pretrained

- Training loss giảm đều qua các epoch, cho thấy mô hình đang học tốt trên tập huấn luyện.
- Validation loss giảm mạnh trong 5 epoch đầu, sau đó gần như không giảm thêm, dao động quanh giá trị ổn định → đây là dấu hiệu mô hình đã hội tụ sớm trên tập kiểm định.
- Sau epoch thứ 5, khoảng cách giữa training và validation loss bắt đầu giãn ra, tuy chưa rõ rệt nhưng có thể là dấu hiệu sớm của overfitting nếu tiếp tục huấn luyện thêm.

- Điều này phản ánh mô hình pretrained đã học được biểu diễn ngôn ngữ tốt, và fine-tuning chủ yếu giúp thích nghi với dữ liệu cụ thể. Sự chênh lệch nhỏ giữa hai đường loss chưa cho thấy hiện tượng overfitting nghiêm trọng.

3.2 Đánh giá BLEU/ROUGE:

Bảng 3. Đánh giá BLEU/ROUGE

Mô hình	BLEU	ROUGE-1	ROUGE-L
Transformer từ đầu	17.8661	0.6367	0.5116
Transformer có pretrained	50.17	0.7421	0.6700
GPT có pretrained (GPT2)	14.53	0.5495	0.47664

Bảng 3 cung cấp kết quả đánh giá hiệu suất của ba mô hình dịch máy (Transformer từ đầu, Transformer có pretrained, và GPT có pretrained - GPT-2) dựa trên các chỉ số BLEU, ROUGE-1 và ROUGE-L. Dưới đây là nhận xét chi tiết:

- Transformer từ đầu:
  + Đạt BLEU là 17.8661, ROUGE-1 là 0.6367 và ROUGE-L là 0.5116.
  + Hiệu suất của mô hình này nằm ở mức trung bình so với các mô hình khác. Chỉ số BLEU cho thấy khả năng tái tạo n-gram chính xác ở mức chấp nhận được, trong khi ROUGE-1 và ROUGE-L phản ánh mức độ bao phủ từ vựng và cấu trúc câu ở mức khá, nhưng vẫn còn khoảng cách so với các mô hình pretrained.
- Transformer có pretrained:
  + Ghi nhận BLEU là 50.17, ROUGE-1 là 0.7421 và ROUGE-L là 0.6700.
  + Đây là mô hình có hiệu suất cao nhất trong cả ba chỉ số. BLEU cao (50.17) cho thấy khả năng dịch chính xác và tự nhiên vượt trội, trong khi ROUGE-1 và ROUGE-L cao (lần lượt 0.7421 và 0.6700) thể hiện sự bao phủ từ vựng và cấu trúc câu rất tốt. Việc sử dụng pretrained rõ ràng mang lại lợi thế lớn về chất lượng dịch.

- GPT có pretrained (GPT-2):
  + Đạt BLEU là 14.53, ROUGE-1 là 0.5495 và ROUGE-L là 0.47664.
  + Mô hình này có hiệu suất thấp nhất trong cả ba chỉ số. BLEU chỉ đạt 14.53, thấp hơn cả Transformer từ đầu, cho thấy khả năng tái tạo n-gram chưa tốt. ROUGE-1 và ROUGE-L cũng thấp (0.5495 và 0.47664), phản ánh mức độ bao phủ từ vựng và cấu trúc câu hạn chế. Điều này có thể do GPT-2 không được tối ưu hóa đặc biệt cho nhiệm vụ dịch máy như Transformer.

Transformer có pretrained vượt trội rõ rệt so với hai mô hình còn lại, chứng minh lợi ích của việc sử dụng trọng số pretrained trong dịch máy. Transformer từ đầu tuy không có pretrained nhưng vẫn đạt kết quả khả quan, vượt qua GPT-2. GPT-2, dù có pretrained, lại không phù hợp lắm cho nhiệm vụ dịch máy, có thể do kiến trúc và cách huấn luyện ban đầu của nó thiên về sinh văn bản hơn là dịch thuật.

TÀI LIỆU THAM KHẢO

Tiếng Việt

…

Tiếng Anh

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. https://doi.org/10.48550/arXiv.1706.03762