Journal of Science and Transport Technology

University of Transport Technology

Introduction to DetectGPT - a model to detect machine-generated text

Nam Hoang Le
Ha Noi University, Ha Noi, Viet Nam

Article info
Type of article:
Original research paper

Corresponding author:
E-mail address:
namlh@hanu.edu.vn

Received: 10/11/2022
Accepted: 27/12/2022
Published: 17/03/2023

Abtract: With the explosion of Large language models (LLMs), recent advances in natural language generation using artificial intelligence are creating a wave of opinions on social networking and artificial intelligence community. Especially, with the recent launch of ChatGPT, young people, specifically students, have begun to update and apply this model in the field of learning. With ChatGPT or similar large language models, students can complete essays with very little effort and attention. This is also a problem in the teaching and learning process of universities. Since then, a group of researchers at Stanford University have proposed a model, called DetectGPT, for the purpose of detecting texts generated by ChatGPT. The article will analyze the DetectGPT system model from an algorithmic perspective and compare the results of DetectGPT with existing models. Then, highlight the outstanding features of the DetectGPT model.

Keyword: DetectGPT, log probability, artificial intelligent, machine-generated text, zero-shot.

.

Tạp chí điện tử
Khoa học và Công nghệ Giao thông
Trường Đại học Công nghệ GTVT

Giới thiệu về DetectGPT – Công cụ phát hiện văn bản do máy tính tạo ra

Lê Hoàng Nam
Trường Đại học Hà Nội, Hà Nội, Việt Nam

Thông tin bài viết
Dạng bài viết:
Bài báo nghiên cứu

***Tác giả liên hệ:**
Địa chỉ E-mail:
namlh@hanu.edu.vn

Ngày nộp bài: 10/11/2022
Ngày chấp nhận: 27/12/2022
Ngày đăng bài: 17/03/2023

Tóm tắt: Với sự bùng nổ của những mô hình ngôn ngữ lớn - Large language models (LLMs), những tiến bộ gần đây trong việc tạo ngôn ngữ tự nhiên bằng trí tuệ nhân tạo đang tạo ra một làn sóng ý kiến trên các trang mạng xã hội và cộng đồng trí tuệ nhân tạo. Đặc biệt, với sự ra mắt gần đây của ChatGPT, giới trẻ, cụ thể là các sinh viên đã bắt đầu cập nhập và ứng dụng mô hình này trong lĩnh vực học tập. Với ChatGPT hoặc các mô hình ngôn ngữ lớn tương tự, sinh viên hoàn toàn có thể hoàn thành các bài tập tiểu luận với rất ít công sức và sự chú tâm. Điều này cũng rấy lên những dư ngại trong quá trình dạy và học của các trường đại học. Từ đó, một nhóm các nhà nghiên cứu tại trường đại học Stanford đã đề xuất một mô hình, có tên gọi DetectGPT, với mục đích phát hiện các văn bản được tạo ra bởi ChatGPT. Bài báo sẽ phân tích mô hình hệ thống DetectGPT dưới góc nhìn thuật toán và so sánh kết quả của DetectGPT với các mô hình đã tồn tại. Rồi từ đó nêu lên được đặc trưng nổi trội của mô hình DetectGPT.

Từ khóa: Trí tuệ nhân tạo, văn bản do máy tạo ra, xác suất logarit.

I. ĐẶT VẤN ĐỀ

Việc sử dụng công cụ trí tuệ nhân tạo để viết nội dung đang có dấu hiệu tăng vọt trong thời gian gần đây. Trong đó, cái tên nổi bật nhất là ChatGPT, được thành lập bởi OpenAI, công cụ chatbot được rất nhiều người yêu thích. Công cụ này có ứng dụng trong rất nhiều lĩnh vực: từ lập trình viên, sáng tạo nội dung, báo chí, nhân viên văn phòng đến những người yêu thích công nghệ. Song song với những mặt tích cực, các công cụ trí tuệ nhân tạo cũng đặt ra vấn đề quan ngại trong vấn đề lạm dụng công cụ trong bài thi của học sinh, sinh viên. Trên thực tế, trường đại học RV ở Bengaluru Ấn Độ đã hoàn toàn nghiêm cấm sinh viên sử dụng ChatGPT với mục đích học thuật như thi cử, thực hành lab và các bài luận [1]. Việc sử dụng các công cụ trí tuệ nhân tạo để viết nội dung trong quá trình làm bài tập, đặc biệt là bài tập luận dưới dạng viết, không chỉ tạo nên sự không công bằng trong việc học đối với người học mà còn tạo nên những bài luận với kiến thức sai lệch. Tuy các mô hình LLMs đã đạt thành công nhất định nhưng độ chính xác về nội dung (đặc biệt là những kiến thức chuyên sâu trong chuyên ngành) chưa được cao [2].

Việc nhận biết một văn bản do con người viết hay do máy tạo ra dựa trên các khuôn mẫu rất khó đối với con người [3], vì vậy các nhà khoa học đã nghiên cứu các phương pháp phân loại tự động để có thể xác định được các khuôn mẫu này. Những phương pháp như vậy có thể giúp giáo viên và người đọc tin tức tin tưởng hơn vào nguồn gốc của văn bản mà họ sử dụng.

Bài báo này sẽ phân tích và tổng hợp nội dung từ nghiên cứu của nhóm các nhà khoa học trường Đại học Stanford: Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, Chelsea Finn với bài nghiên cứu mang tiêu đề: DetectGPT: Zero-Shot Machine-

Generated Text Detection using Probability Curvature [4]. Trong nghiên cứu này, DetectGPT được ra đời để nhận diện văn bản sinh ra bởi máy bằng cách vận dụng các đặc tính riêng biệt của các văn bản sinh ra bởi mô hình ngôn ngữ. Để bám sát nội dung bài nghiên cứu gốc, một số từ ngữ tiếng Anh chuyên ngành sẽ được giữ nguyên (có đi kèm bản dịch tương đối).

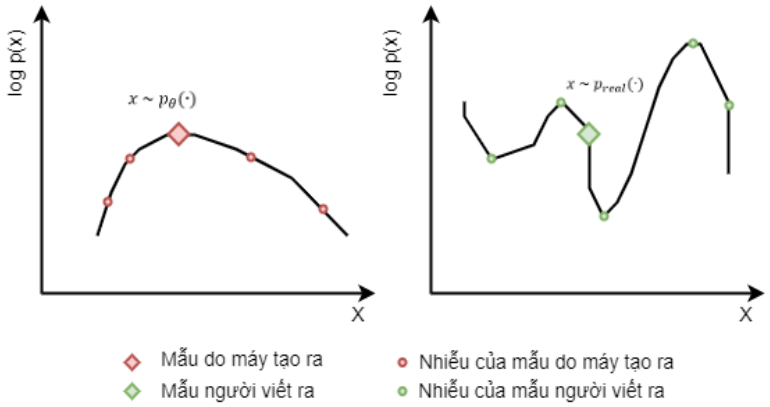
Về nội dung bài báo, sau phần đặt vấn đề, phần 0 sẽ trình bày về hàm xác suất logarit và đường cong xác suất; tổng quát phương thức hoạt động của mô hình DetectGPT. Tiếp đó, phần 0 sẽ giới thiệu cơ sở dữ liệu và chỉ số đo lường. Cuối cùng, so sánh kết quả với các mô hình đã tồn tại để thấy được hiệu quả của mô hình DetectGPT ở phần 0.

II. PHƯƠNG PHÁP NGHIÊN CỨU

1. MỐI QUAN HỆ GIỮA HÀM XÁC SUẤT LOGARIT VÀ CÁC VĂN BẢN DO LLMs TẠO RA

Để giải quyết bài toán phát hiện những văn bản do máy tính tạo ra, chúng ta đặt ra một giả thuyết đơn giản và sau đó chứng minh giả thuyết này bằng thực nghiệm. Giả thuyết đề ra là: các bản viết lại do mô hình máy tính tạo ra có xu hướng xác suất logarit thấp hơn so với mẫu ban đầu, trong khi các bản viết lại do con người viết có thể có khả năng xác suất logarit cao hơn hoặc thấp hơn so với bản gốc. Nói cách khác, không giống như văn bản do con người viết, văn bản do mô hình tạo có xu hướng nằm trong các khu vực mà hàm xác suất logarit có độ cong âm (ví dụ: điểm cực đại cục bộ của xác suất logarit).

Ta có thể thấy giả thuyết này được biểu diễn dưới biểu đồ Biểu đồ.



Biểu đồ 1. Biểu đồ đường cong xu hướng của xác suất logarit của văn bản do máy tính tạo và văn bản do con người viết

Biểu đồ xác định và khai thác xu hướng của các đoạn do máy tạo $x \sim p_\theta(\cdot)$ (bên trái) nằm trong vùng cong âm của log p(x), trong đó các mẫu gần đó có xác suất trung bìn logarit thấp hơn. Ngược lại, văn bản do con người viết $x \sim p_{real}(\cdot)$ (bên phải) có xu hướng không chiếm các vùng có độ cong xác suất logarit âm rõ ràng.

Từ đó, chúng ta có thể xác định và xác thực giả thuyết rằng độ cong của hàm xác suất logarit của mô hình máy tính có xu hướng âm hơn so với văn bản của con người.

Cụ thể, để kiểm tra xem một đoạn văn có đến từ một mô hình nguồn p_θ hay không, DetectGPT so sánh xác suất logarit của một đoạn ứng viên trong p_θ với xác suất logarit trung bình của một số nhiễu loạn (perturbations) của đoạn trong p_θ (ví dụ: T5 [5]). Nếu các đoạn nhiễu loạn có xu hướng có xác suất logarit trung bình thấp hơn so với bản gốc, thì đoạn ứng cử viên có khả năng đến từ p_θ.

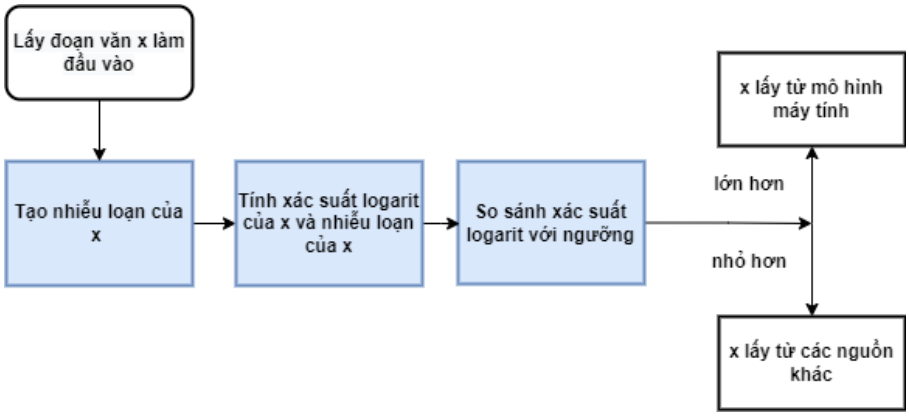
Biểu đồ 2 trình bày tổng quát bài toán và mô hình DetectGPT.

Mục đích của bài toán là xác định xem một đoạn văn bản có được tạo bởi một LLM p cụ thể hay không, chẳng hạn như GPT-3. Để phân loại một đoạn ứng cử viên x, trước tiên, DetectGPT tạo ra các nhiễu loạn nhỏ của đoạn $\tilde{x}_i$ bằng cách sử dụng một mô hình được đào tạo trước chẳng hạn như T5 [5]. Sau đó, DetectGPT so sánh xác suất

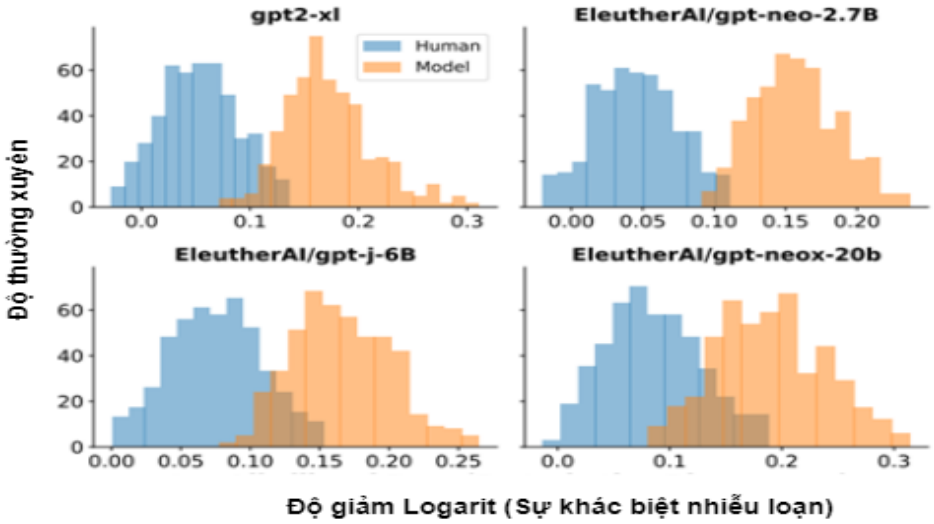
logarit theo p của mẫu ban đầu x với từng mẫu bị nhiễu $\tilde{x}_i$. Nếu tỷ lệ logarit trung bình cao thì mẫu văn bản có khả năng từ mô hình nguồn.

Sự trung bình giảm của xác suất logarit (chênh lệch nhiễu loạn) sau khi viết lại một đoạn văn do mô hình máy tính tạo ra luôn cao hơn đối với các đoạn văn do con người viết. Mỗi biểu đồ cho thấy sự phân bố của chênh lệch nhiễu loạn d (x, p_θ, q) đối với các bài báo do con người viết và các bài báo do máy tạo ra; có độ dài từ bằng nhau từ các mẫu GPT-2 (1.5B), GPT-Neo2 (7B [6]), GPT-J (6B [7]) và GPT-NeoX (20B [8]). Các bài báo do con người viết là một mẫu gồm 500 bài báo XSum; văn bản do máy tạo được tạo bằng cách nhắc từng mô hình với 30 mã tokens đầu tiên của mỗi bài viết XSum, lấy mẫu từ điều kiện thô.

Biểu đồ 3 đánh giá thực nghiệm của giả thuyết.



Biểu đồ 2. Mô hình DetectGPT



Biểu đồ 3. Biểu đồ thể hiện sự trung bình giảm của xác suất logarit của văn bản do máy tính tạo và do con người viết

2. PHƯƠNG THỨC HOẠT ĐỘNG CỦA DETECTGPT

Dựa trên giả thuyết ở phần trước bài báo, mô hình DetectGPT đã được phát triển sử dụng hàm xác suất logarit Hessian để phát hiện những văn bản được tạo ra bởi mô hình máy tính.

Các mô hình giám sát phát hiện văn bản do máy tạo ra đã được phát triển dưới rất nhiều các hình thức khác nhau. Trong khi đó, mô hình DetectGPT được phát triển dưới dạng Zero-shot. Mô hình này sử dụng chính mô hình nguồn mà không cần tinh chỉnh hoặc điều chỉnh dưới bất kỳ hình thức nào. Phương pháp phổ biến nhất là đánh giá xác suất logarit trung bình trên mỗi mã token của văn bản được tạo và ngưỡng (threshold) ([9], [10]). Về cơ bản phương pháp "zero-shot" cho

phép DetectGPT phát hiện văn bản do trí tuệ nhân tạo viết mà không cần biết về loại trí tuệ nhân tạo đã được sử dụng để tạo văn bản đó. Nó hoạt động hoàn toàn trái ngược với các phương pháp kiểm tra khác thường yêu cầu đào tạo ra một 'bộ phân loại' và sử dụng bộ dữ liệu của các đoạn văn bản thật và giả. Tức là DetectGPT không cần huấn luyện hay dữ liệu có nhãn để phát hiện một văn bản là giả mạo mà hoàn toàn có thể tính toán trực tiếp.

DetectGPT dựa trên giả thuyết rằng các mẫu văn bản từ mô hình nguồn p_θ thường nằm trong vùng có độ cong âm của hàm xác suất logarit của p_θ, không giống như văn bản của con người.

Nói cách khác, nếu chúng ta áp dụng các nhiễu loạn nhỏ cho một đoạn $x \sim p_\theta$, tạo ra $\tilde{x}$, đại lượng của $\log p_\theta(x) - \log p_\theta(\tilde{x})$ trung bình sẽ lớn đối với các mẫu do máy tạo ra so với mẫu do con người viết ra chữ. Để tận dụng giả thuyết này, trước tiên hãy xem xét một hàm nhiễu $q(\cdot|x)$ cung cấp phân phối trên $\tilde{x}$, các phiên bản được sửa đổi một chút của x với ý nghĩa tương tự. Ví dụ: $q(\cdot|x)$ có thể là kết quả của việc yêu cầu một người viết lại một trong các câu của x, trong khi vẫn giữ nguyên nghĩa của x. Sử dụng khái niệm hàm nhiễu loạn, chúng ta có thể định nghĩa sai lệch nhiễu loạn $d(x, p_\theta, q)$:

$$d(x, p_\theta, q) \triangleq \log p_\theta(x) - \mathbb{E}_{\tilde{x}\sim q(\cdot|x)} \log p_\theta(\tilde{x}) \quad (1)$$

(Phương trình 1)

Nguyên văn giả thuyết: “If q produces samples on the data manifold, $d(x, p_\theta, q)$ is positive with high probability for samples $x \sim p_\theta$. For human-written text, $d(x, p_\theta, q)$ tends toward zero for all x.” [4]

Nếu chúng ta xác định $q(\cdot|x)$ là mẫu từ mô hình chứ không phải do con người viết lại, thì chúng ta có thể kiểm tra thực nghiệm Giả thuyết. Biểu đồ cho thấy kết quả của thí nghiệm. Nó chỉ ra rằng sự phân bố của sự khác biệt nhiễu loạn (perturbation discrepancy) là khác biệt đáng kể giữa bài báo do con người viết và do mô hình máy tính; các mẫu mô hình máy có xu hướng có sự khác biệt nhiễu loạn lớn hơn.

Với những kết quả này, chúng ta có thể phát hiện xem một đoạn văn bản có được tạo bởi một mô hình p_θ hay không bằng cách tạo ra một ngưỡng cho sự khác biệt nhiễu loạn. Trong thực tế, chúng ta nhận thấy rằng việc chuẩn hóa chênh lệch nhiễu loạn theo độ lệch chuẩn của giá trị quan sát $E_{\tilde{x}\sim q(\cdot|x)} \log p_\theta(\tilde{x})$ cung cấp kết quả tốt hơn, vì vậy chúng ta sử dụng phiên bản chuẩn hóa này trong các thử nghiệm. Cuối cùng, phương pháp DetectGPT được tóm tắt trong thuật toán 1.

1. Đầu vào: Đoạn văn x, mô hình nguồn p_θ, hàm nhiễu loạn p, số lượng nhiễu loạn k, ngưỡng ϵ
2. $\tilde{x}_i \sim q(.|x), i \in [1...k]$
3. $\tilde{\mu} \leftarrow \frac{1}{k}\Sigma_i log p_\theta(\tilde{x}_i)$
4. $\hat{d}_x \leftarrow log p_\theta(x) - \tilde{\mu}$
5. $\tilde{\sigma}_x^2 \leftarrow \frac{1}{k-1}\Sigma_i (log p_\theta(\tilde{x}_i) - \tilde{\mu})^2$
6. **Nếu** $\frac{\hat{d}_x}{\sqrt{\tilde{\sigma}_x}} > \epsilon$ **thì**
7. Trả lại giá trị *true*
8. **Nếu không thì**
9. Trả lại giá trị *false*

(Thuật toán 1)

Giải thích sự khác biệt nhiễu loạn (Perturbation Discrepancy)

Ở phần này, chúng ta sẽ phân tích sự khác biệt nhiễu loạn như là một đơn vị đo lường của đường cong cục bộ của hàm xắc suất logarit. Cụ thể hơn, nó sẽ tỉ lệ thuận với vết âm tính (negative trace) của hàm xắc suất logarit Hessian.

Trước tiên, chúng ta có khái niệm Hutchinson’s trace estimator [11], đưa ra ước tính của ma trận A:

$$\mathbf{tr}(A) = \mathbb{E}_{\mathbf{z}} \mathbf{z}^\top A \mathbf{z} \quad (2)$$

(Phương trình 2)

Trong đó, các phần tử của $z \sim q_z$ là biến độc lập và phân phối đồng nhất (Independent and Identically Distributed) với $E[z_i] = 0$ và $Var(z_i) = 1$.

Để sử dụng phương trình (Phương trình 2 cho vết của Hessian, ta cần tính đạo hàm cấp 2 có hướng của $\mathbf{z}^\top H_f(x)\mathbf{z}$. Chúng ta có một biểu thức với

sự khác biệt hữu hạn:

$$\mathbf{z}^\top H_f(x)\mathbf{z} \approx \frac{f(x+h\mathbf{z}) + f(x-h\mathbf{z}) - 2f(x)}{h^2} \quad (3)$$

(Phương trình 3)

Kết hợp phương trình Phương trình 2 và Phương trình 3 với h = 1, chúng ta có ước lượng của vết âm tính Hessian:

$$-\mathrm{tr}(H)_f(x) \approx 2f(x) - \mathbb{E}_{\mathbf{z}}\left[f(x+\mathbf{z}) + f(x-\mathbf{z})\right]. \quad (4)$$

(Phương trình 4)

Nếu sự phân phối nhiễu (noise distribution) là đối xứng thì p(**z**) = p(-**z**) với tất cả **z**, chúng ta có thể đơn giản hóa phương trình (Phương trình 4) như sau:

$$\frac{-\mathrm{tr}(H)_f(x)}{2} \approx f(x) - \mathbb{E}_{\mathbf{z}} f(x+\mathbf{z}). \quad (5)$$

(Phương trình 5)

Có một lưu ý là vế bên phải của phương trình, Phương trình 5 tương ứng với chênh lệch nhiễu loạn, Phương trình 1 với hàm nhiễu loạn $q(\tilde{x}|x)$ được thay thể bởi $q_z(z)$ trong vết Hutchinson Phương trình 2. Ở đây, $\tilde{x}$ là một chuỗi mã token nhiều chiều trong khi q_z là một vectơ trong không gian ngữ nghĩa nhỏ. Do mô hình lấy mẫu các câu tương tự như x với rất ít thay đổi về mặt ngữ nghĩa, nên chúng ta có thể coi mô hình như một phép nhúng từ (embedding) ngữ nghĩa tương tự ($\tilde{z} \sim q_z$) và sau đó liên kết mô hình này tới một chuỗi token ($\tilde{z} \rightarrow \tilde{x}$). Việc lấy mẫu trong không gian ngữ nghĩa đảm bảo rằng tất cả các mẫu đều trong vùng dữ liệu đa dạng. Do đó, chúng ta có thể giải thích mục tiêu của bài toán xấp xỉ độ cong bị giới hạn với dữ liệu đa dạng.

III. DỮ LIỆU

Cơ sở dữ liệu và chỉ số đo lường

Mô hình sử dụng 3 bộ dữ liệu từ rất nhiều nguồn LLMs: Các bài báo từ bộ dữ liệu XSum [12], bộ dữ liệu các đoạn Wikipedia từ SquAD [13] và các câu chuyện từ bộ dữ liệu Reddit WritingPrompts [14].

Bộ dữ liệu XSum đại diện cho việc phát hiện tin tức giả mạo, SquAD đại diện cho các bài luận học thuật do máy viết và WritingPrompts đại diện các bài viết sáng tạo do máy tạo ra. Kết quả được thể hiện trong Bảng 1.

Sau đó, mô hình sử dụng tiếp 3 nhóm dữ liệu tiếp theo để đánh giá sự thay đổi phân phối (distribution shift). Sự thay đổi phân phối xảy ra khi việc phân phối dữ liệu mà mô hình nhìn thấy trong quá trình sản xuất bắt đầu trông khác với dữ liệu được sử dụng để huấn luyện nó. Bộ 3 dữ liệu bao gồm: Cơ sở dữ liệu phiên bản tiếng Anh của WMT16 [15], cơ sở dữ liệu phiên bản tiếng Đức của WMT16 [15] và cơ sở dữ liệu các câu trả lời dạng dài được viết bởi các chuyên gia về con người trong bộ dữ liệu PubMedQA [16]. Kết quả được thể hiện trong Biểu đồ 4.

Chỉ số được sử dụng trong bài viết để đo lường hiệu suất của mô hình là Area Under the Receiver Operating Characteristic Curve (AUROC), đơn vị này có thể hiểu là xác suất mà bộ phân loại xếp hạng chính xác giá trị dương được chọn ngẫu nhiên (do máy tạo) cao hơn giá trị âm được chọn ngẫu nhiên (do con người viết).

IV. KẾT QUẢ VÀ THẢO LUẬN

So sánh với các phương pháp Zero-shot khác

Ở phần này, bài viết sẽ so sánh DetectGPT với các phương pháp zero-shot hiện có khác nhau để phát hiện văn bản do máy tạo. Các phương pháp này tương ứng với các thử nghiệm thống kê dựa trên xác suất logarit mã token, xếp hạng mã token hoặc entropy dự đoán ([9], [10]).

Phương pháp đầu tiên (log p(x)) sử dụng xác suất logarit trung bình mã token của mô hình nguồn để xác định xem một đoạn ứng viên có phải do máy tạo ra hay không; các đoạn có xác suất logarit trung bình cao có khả năng được tạo bởi mô hình máy tính.

Phương pháp thứ hai (Rank) và thứ ba (LogRank) sử dụng thứ hạng trung bình hoặc thứ hạng logarit của mã token trong đoạn ứng viên theo phân phối có điều kiện của mô hình. Các đoạn văn có thứ hạng trung bình (log-) nhỏ hơn có khả năng là do máy tạo ra.

Phương pháp cuối cùng (Entropy) dựa trên giả thuyết Gehrmann [9]. Entropy dự đoán có liên quan tích cực với tính giả tạo của đoạn văn; do đó, đường cơ sở (baseline) này sử dụng entropy trung bình cao trong phân phối dự đoán của mô hình làm tín hiệu cho thấy một đoạn văn do máy tạo ra.

Từ Bảng 1, AUROC của DetectGPT và bốn tiêu chí được đề xuất trước đó để phát hiện các mẫu từ mô hình nhất định trên tập dữ liệu đã cho (500 mẫu được sử dụng để đánh giá). Chỉ số AUC càng cao thì mô hình càng chính xác trong việc phân loại các lớp. Số liệu in đậm hiển thị AUROC tốt nhất trong mỗi cột; số liệu có dấu hoa thị biểu thị AUROC tốt thứ hai. Các giá trị ở hàng cuối cùng hiển thị AUROC của DetectGPT so với phương pháp cơ sở mạnh nhất trong cột đó. Trong thử nghiệm này, DetectGPT vượt trội so với đường cơ sở zero-shot mạnh nhất, cụ thể là hơn 0,1 AUROC trên XSum và 0,05 AUROC trên ngữ cảnh SQuAD Wikipedia.

So sánh với các phương pháp phát hiện giám sát

Biểu đồ 4 thể hiện sự so sánh giữa phương pháp zero-shot của DetectGPT và các phương pháp phát hiện giám sát khác. Các phương pháp giám sát này chủ yếu sử dụng biến đổi tinh chỉnh (fine-tune).

Ta có thể thấy, các mô hình giám sát phát hiện văn bản do máy tạo được đào tạo trên bộ dữ liệu gồm văn bản thực và văn bản được tạo hoạt động tốt bằng hoặc tốt hơn so với DetectGPT trên văn bản phân phối (hàng trên cùng). Tuy nhiên, các phương pháp zero-shot hoạt động vượt trội đối với các miền mới (hàng dưới cùng) chẳng hạn như văn bản y tế PubMed và dữ liệu tin tức tiếng Đức từ WMT16. DetectGPT có hiệu suất rất cạnh tranh so với các mô hình giám sát hiện hành.

Thay đổi Chiến lược giải mã

Mặc dù Bảng 1 cho thấy rằng rằng DetectGPT hiệu quả để phát hiện văn bản do máy tạo, nhưng chiến lược giải mã (Decoding Strategy) (lấy mẫu top-*k*, top-*p*) có thể ảnh hưởng đến độ khó của phát hiện. Ở phần này, chúng ta sẽ sử dụng lấy mẫu top-*k* và lấy mẫu hạt nhân top-*p*. Lấy mẫu top-*k* cắt bớt lấy mẫu phân phối thành *k* mã token tiếp theo có xác suất cao nhất; các mẫu lấy mẫu hạt nhân top-*p* chỉ tập hợp các kết quả tokens có xác suất kết hợp vượt không vượt quá *p*. Kết quả của thí nghiệm được tóm tắt trong các bảng Bảng 2; Bảng 3 và Bảng 4. Mô hình sử dụng k = 40 và p = 0,96, hai chỉ số này cũng đã từng được sử dụng trong bài nghiên cứu trước đó của Ippolito [10]. Ở thí nghiệm này, kết quả cho thấy DetectGPT cung cấp hiệu quả rõ ràng nhất trong cả chỉ số top-*k* và top-*p.*

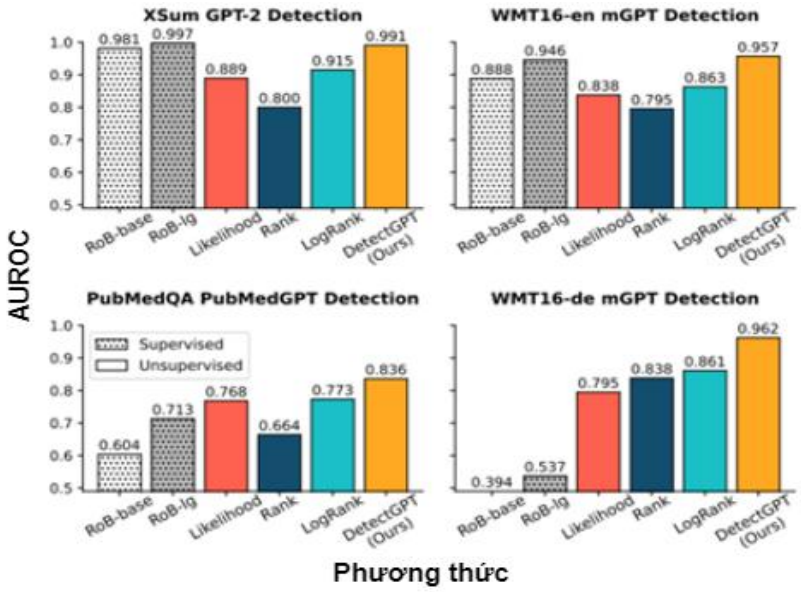
AUROC cho các phương pháp zero-shot lấy trung bình trên năm mô hình trong Bảng 1 cho cả lấy mẫu top-*k* và top-*p* (với k = 40 và p = 0,96). Cả hai cài đặt đều cho phép phát hiện chính xác hơn và DetectGPT luôn cung cấp hiệu suất phát hiện tốt nhất.

Cụ thể ở Bảng 3, AUROC để phát hiện các mẫu từ mô hình trên bộ dữ liệu đã cho cho DetectGPT và bốn tiêu chí được đề xuất trước đó. Nhìn chung, cách thức lấy mẫu hạt nhân làm cho tất cả các phương pháp phát hiện dễ dàng hơn, DetectGPT vẫn cung cấp AUROC trung bình cao nhất. Chỉ riêng với WritingPrompts, đường cơ sở LogRank hoạt động tốt hơn DetectGPT.

Với Bảng 4, DetectGPT thường cung cấp hiệu suất chính xác nhất (AUROC cao nhất), mặc dù khoảng cách được thu hẹp so với lấy mẫu trực tiếp.

Bảng 1. Bảng so sánh DetectGPT và bốn tiêu chí đánh giá

	XSum						SQuAD						WritingPrompts					
Method	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**
$\log p(x)$	0.86	0.86	0.86	0.82	0.77	0.83	0.91	0.88	0.84	0.78	0.71	0.82	0.97	0.95	0.95	0.94	0.93*	0.95
Rank	0.79	0.76	0.77	0.75	0.73	0.76	0.83	0.82	0.80	0.79	0.74	0.80	0.87	0.83	0.82	0.83	0.81	0.83
LogRank	0.89*	0.88*	0.90*	0.86*	0.81*	0.87*	0.94*	0.92*	0.90*	0.83*	0.76*	0.87*	0.98*	0.96*	0.97*	0.96*	**0.95**	0.96*
Entropy	0.60	0.50	0.58	0.58	0.61	0.57	0.58	0.53	0.58	0.58	0.59	0.57	0.37	0.42	0.34	0.36	0.39	0.38
DetectGPT	**0.99**	**0.97**	**0.99**	**0.97**	**0.95**	**0.97**	**0.99**	**0.97**	**0.97**	**0.90**	**0.79**	**0.92**	**0.99**	**0.99**	**0.99**	**0.97**	0.93*	**0.97**
Diff	0.10	0.09	0.09	0.11	0.14	0.10	0.05	0.05	0.07	0.07	0.03	0.05	0.01	0.03	0.02	0.01	-0.02	0.01



Biểu đồ 1. Bảng so sánh phương pháp DetectGPT và các phương pháp phát hiện giám sát khác

Bảng 2. Đánh giá AUROC giữa các phương pháp Zero-shot với lấy mẫu top-k và top-p

	XSum		SQuAD		WritingPrompts	
Method	top-p	top-k	top-p	top-k	top-p	top-k
$\log p(x)$	0.92	0.87	0.89	0.85	**0.98**	0.96
Rank	0.76	0.76	0.81	0.80	0.84	0.83
LogRank	0.93*	0.90*	0.92*	0.90*	**0.98**	**0.97**
Entropy	0.53	0.55	0.54	0.56	0.32	0.35
DetectGPT	**0.98**	**0.98**	**0.94**	**0.93**	**0.98**	**0.97**

Bảng 3. Bảng đánh giá phương pháp lấy mẫu Nucleus (top-p) với p = 0.96

	XSum						SQuAD						WritingPrompts					
Method	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**
$\log p(x)$	0.93	0.93	0.94	0.91	0.87	0.92	0.96	0.94	0.91	0.87	0.79	0.89	0.99*	0.98*	0.98*	0.97*	0.97*	**0.98**
Rank	0.80	0.77	0.77	0.75	0.73	0.76	0.84	0.82	0.81	0.80	0.75	0.81	0.87	0.84	0.83	0.83	0.81	0.84
LogRank	0.95*	0.94*	0.96*	0.93*	0.89*	0.93*	0.98*	0.96*	0.94*	**0.90**	**0.83**	0.92*	0.99*	0.98*	0.98*	**0.98**	**0.98**	**0.98**
Entropy	0.55	0.46	0.53	0.54	0.58	0.53	0.53	0.50	0.55	0.56	0.57	0.54	0.32	0.37	0.28	0.32	0.32	0.32
DetectGPT	**0.99**	**0.98**	**1.00**	**0.98**	**0.97**	**0.98**	**0.99**	**0.98**	**0.98**	**0.90**	0.82*	**0.94**	**1.00**	**0.99**	**0.99**	0.97*	0.93	**0.98**
Diff	0.04	0.04	0.04	0.05	0.08	0.05	0.01	0.02	0.04	0.00	-0.01	0.02	0.01	0.01	0.01	-0.01	-0.05	0.00

Bảng 4. Bảng đánh giá phương pháp lấy mẫu Top-k với k = 40

	XSum						SQuAD						WritingPrompts					
Method	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**	GPT-2	OPT-2.7	Neo-2.7	GPT-J	NeoX	**Avg.**
$\log p(x)$	0.89	0.89	0.89	0.84	0.81	0.87	0.93	0.90	0.88	0.82	0.74	0.85	0.97	0.95	0.97	0.96	0.95*	0.96
Rank	0.79	0.77	0.77	0.75	0.73	0.76	0.84	0.82	0.80	0.80	0.75	0.80	0.87	0.84	0.83	0.82	0.81	0.83
LogRank	0.92*	0.91*	0.93*	0.89*	0.85*	0.90*	0.96*	0.94*	0.92*	0.87*	0.79*	0.90*	0.98*	0.97*	0.98*	**0.97**	**0.96**	**0.97**
Entropy	0.58	0.49	0.55	0.56	0.59	0.55	0.55	0.52	0.56	0.56	0.58	0.56	0.35	0.41	0.30	0.34	0.37	0.35
DetectGPT	**0.99**	**0.97**	**0.99**	**0.96**	**0.96**	**0.98**	**0.99**	**0.98**	**0.98**	**0.89**	**0.80**	**0.93**	**0.99**	**0.98**	**0.99**	**0.97**	0.93	**0.97**
Diff	0.07	0.06	0.06	0.07	0.11	0.08	0.03	0.04	0.06	0.02	0.01	0.03	0.01	0.01	0.01	0.00	-0.03	0.00

V. KẾT LUẬN

Song song với sự phát triển của mô hình tạo ra văn bản bằng máy tính (GPT-2, GPT-3 và gần đây là ChatGPT), thì mô hình phát hiện văn bản tạo ra bằng máy tính cũng cần phải phát triển theo. Hệ thống giáo dục cần chuẩn bị kĩ cho vấn đề này trong tương lai dưới sự bùng nổ của trí tuệ nhân tạo trong việc học sinh, sinh viên sử dụng trí tuệ nhân tạo một cách lạm dụng trong các bài thi. Thậm chí, các cơ sở giáo dục có thể yêu cầu người học sử dụng các công cụ phát hiện văn bản do máy tính tạo ra nếu cần thiết.

DetectGPT đã đưa ra được một môt hình có tính thực nghiệm cao, nhằm làm bước nền để cộng đồng trí tuệ nhân tạo có thể phát triển nên các mô hình có độ chính xác cao hơn. Sau khi so sánh DetectGPT với các phương pháp Zero-shot hiện có, khả năng phân biệt của DetectGPT cao đáng kể. Đồng thời, mô hình này còn chỉ ra rất nhiều đặc điểm và phương hướng phát triển trong tương lai. Hiện tại, mô hình DetectGPT chưa công bố mã nguồn mở nhưng mô hình được thông báo sẽ công bố mã nguồn cho cộng đồng trí tuệ nhân tạo trong thời gian sớm nhất. Để tổng kết, bài báo này tập trung vào phân tích cách thức hoạt động của mô hình DetectGPT và những kết quả chính đạt được từ phương thức này, mang lại cho người đọc một cái nhìn tổng quát với hệ thống phát hiện văn bản do máy tính tạo ra.

TÀI LIỆU THAM KHẢO

[1] Hindustantimes News. Why this Bengaluru institute has restricted ChatGPT use for students. Bengaluru news. Published on Jan 28, 2023

[2] Christian, J. CNET secretly used AI on articles that didn’t disclose that fact, staff say. 2023.

[3] Gehrmann, S., Strobelt, H., and Rush, A. GLTR: Statistical detection and visualization of generated text. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 111–116, Florence, Italy, July 2019. Association for Computational Linguistics.

[4] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, Chelsea Finn. (2023). DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. arXiv preprint arXiv:2301.11305.

[5] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140):1–67, 2020.

[6] Black, S., Gao, L., Wang, P., Leahy, C., and Biderman, S. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021.

[7] Wang, B. and Komatsuzaki, A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model

[8] Black, S., Biderman, S., Hallahan, E., Anthony, Q., Gao, L., Golding, L., He, H., Leahy, C., McDonell, K., Phang, J., Pieler, M., Prashanth, U. S., Purohit, S., Reynolds, L., Tow, J., Wang, B., and Weinbach, S. GPT-NeoX-20B: An open-source autoregressive language model. In Proceedings of the ACL Workshop on Challenges & Perspectives in Creating Large Language Models, 2022.

[9] Gehrmann, S., Strobelt, H., and Rush, A. GLTR: Statistical detection and visualization of generated text. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 111–116, Florence, Italy, July 2019. Association for Computational Linguistics.

[10] Ippolito, D., Duckworth, D., Callison-Burch, C., and Eck, D. Automatic detection of generated text is easiest when humans are fooled. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1808–1822, Online, July 2020. Association for Computational Linguistics.

[11] Hutchinson, M. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. Communications in Statistics - Simulation and Computation, 1990

[12] Narayan, S., Cohen, S. B., and Lapata, M. Don’t give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 2018

[13] Rajpurkar, P., Zhang, J., Lopyrev, K., and

Liang, P. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2383– 2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[14] Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neural story generation. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 889–898, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[15] Bojar, O. r., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Jimeno Yepes, A., Koehn, P., Logacheva, V., Monz, C., Negri, M., Neveol, A., Neves, M., Popel, M., Post, M., Rubino, R., Scarton, C., Specia, Marco Turchi, et al. 2016. Findings of the 2016 Conference on Machine Translation. In Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

[16] Jin, Q., Dhingra, B., Liu, Z., Cohen, W., and Lu, X. PubMedQA: A dataset for biomedical research question answering. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2567–2577, Hong Kong, China, November 2019. Association for Computational Linguistics.