THÔNG TIN CHUNG CỦA NHÓM

- Link YouTube video của báo cáo (tối đa 5 phút): *https://www.youtube.com/watch?v=lfZrFTXAak0*
- Link slides (dạng .pdf đặt trên Github của nhóm): *https://github.com/minhquan257/CS2205/blob/main/CS2205.SEP2025.DeCuong.FinalReport.AIO.Slide.pdf*
- *Mỗi thành viên của nhóm điền thông tin vào một dòng theo mẫu bên dưới*
- *Sau đó điền vào Đề cương nghiên cứu (tối đa 5 trang), rồi chọn Turn in*
- *Lớp Cao học, mỗi nhóm một thành viên*

- Họ và Tên: Nguyễn Minh Quân
- MSHV: 250202019

- Lớp: CS2205.RM
- Tự đánh giá (điểm tổng kết môn): 9/10
- Số buổi vắng: 0
- Số câu hỏi QT cá nhân: 7
- Số câu hỏi QT của cả nhóm: 7
- Link Github: https://github.com/mynameuit/CS2205.xxx/

ĐỀ CƯƠNG NGHIÊN CỨU

TÊN ĐỀ TÀI (IN HOA)

SMOOTH-GUARD: GIẢI PHÁP TỐI ƯU KHẢ NĂNG PHÒNG THỦ CHỐNG PROMPT INJECTION CHO CÁC MÔ HÌNH LLM.

TÊN ĐỀ TÀI TIẾNG ANH (IN HOA)

SMOOTH-GUARD: ENHANCING ROBUSTNESS AGAINST PROMPT INJECTION ATTACKS IN LARGE LANGUAGE MODELS.

TÓM TẮT *(Tối đa 400 từ)*

Trong bối cảnh các mô hình ngôn ngữ lớn (LLM) đang trở thành hạ tầng cốt lõi cho các ứng dụng doanh nghiệp, nguy cơ từ các cuộc tấn công Prompt Injection đã bộc lộ những lỗ hổng bảo mật nghiêm trọng. Các phương pháp phòng thủ hiện nay, chẳng hạn như lọc từ khóa dựa trên danh sách đen (Blacklisting), sử dụng các bộ phân loại tĩnh (Static Classifiers), hay huấn luyện căn chỉnh (RLHF), thường tỏ ra kém hiệu quả hoặc dễ bị vượt qua bởi các kỹ thuật xáo trộn từ ngữ (obfuscation) và các biến thể mã độc mới. Để khắc phục những điểm yếu này, chúng tôi đề xuất ứng dụng thuật toán SMOOTHLLM, một giải pháp phòng thủ dựa trên cơ chế làm mịn ngẫu nhiên (randomized smoothing).

Khác với các phương pháp truyền thống vốn dựa vào việc nhận diện các mẫu (pattern) cố định—thứ vốn rất dễ bị bẻ gãy bởi các cuộc tấn công thích ứng—SMOOTHLLM khai thác tính chất 'mỏng manh' về cấu trúc của các câu lệnh chèn độc hại. Bằng cách thực hiện làm nhiễu ngẫu nhiên nhiều phiên bản của prompt đầu vào và tổng hợp phản hồi, SMOOTHLLM có thể phát hiện các

hành vi tấn công mà không cần biết trước mẫu mã độc. Kết quả thực nghiệm cho thấy SMOOTHLLM vượt trội hơn hẳn các giải pháp dựa trên quy tắc (rule-based) và các màng lọc tĩnh trong việc ngăn chặn các cuộc tấn công phức tạp, đồng thời duy trì sự ổn định cho hệ thống mà không đòi hỏi chi phí tái huấn luyện tốn kém. Đây chính là lớp bảo mật linh hoạt và mạnh mẽ, đáp ứng nhu cầu cấp thiết cho việc vận hành LLM an toàn trong thực tế.

GIỚI THIỆU *(Tối đa 1 trang A4)*

1. Bối cảnh và Tính thời sự

Trong giai đoạn 2024-2025, các mô hình ngôn ngữ lớn (LLM) như GPT-4, Llama và Claude đã trở thành hạ tầng công nghệ thiết yếu cho các doanh nghiệp toàn cầu. Tuy nhiên, đi kèm với sự phát triển này là sự gia tăng của lỗ hổng Prompt Injection – được tổ chức OWASP xếp hạng là rủi ro an ninh hàng đầu cho các ứng dụng LLM. Các phương pháp phòng thủ truyền thống như bộ lọc từ khóa tĩnh (Static Filtering) hay huấn luyện căn chỉnh (RLHF) hiện đang bộc lộ điểm yếu trước các kỹ thuật tấn công tinh vi (như GCG, PAIR), vốn có khả năng ngụy trang mã độc dưới dạng các chuỗi ký tự phức tạp. Việc tìm kiếm một giải pháp phòng thủ mạnh mẽ, linh hoạt và không đòi hỏi chi phí tái huấn luyện tốn kém là nhu cầu cấp thiết hiện nay.

2. Xác định Bài toán

Đề tài tập trung giải quyết bài toán Phát hiện đối kháng (Adversarial Detection) trong không gian văn bản rời rạc.

- Bài toán: Xác định tính hợp lệ của một chuỗi đầu vào P thông qua việc đo lường độ ổn định của phản hồi mô hình khi có tác động nhiễu.
- Input: Một câu lệnh (Prompt) P bất kỳ từ người dùng hoặc từ dữ liệu bên thứ ba.
- Output: Quyết định nhị phân $\{0, 1\}$. Trong đó 0 là "Lành tính" (cho phép xử lý) và 1 là "Đối kháng" (ngăn chặn và cảnh báo).

3. Giải pháp đề xuất: Thuật toán SmoothLLM

Nghiên cứu áp dụng thuật toán SmoothLLM, một cơ chế dựa trên nguyên lý Làm mịn ngẫu nhiên (Randomized Smoothing). Giải pháp này được triển khai như một lớp bảo mật Runtime (Lớp tiền xử lý) đứng trước LLM mục tiêu.

Cơ chế hoạt động (Mô tả kỹ thuật):

Dựa trên phát hiện then chốt rằng các câu lệnh tấn công thường có cấu trúc "mỏng manh" (brittle), SmoothLLM thực hiện quy trình 3 bước:

1. Làm nhiễu (Perturbation): Hệ thống tự động tạo ra N bản sao của prompt gốc. Trên mỗi bản sao, thuật toán áp dụng các hàm nhiễu như *Random Swap* (hoán đổi ký tự), *Random Patch* (thay đổi một đoạn ký tự) hoặc *Random Insert* (chèn ký tự ngẫu nhiên).
2. Truy vấn (Inference): Tất cả các bản sao này được đưa qua mô hình LLM cùng một lúc.
3. Tổng hợp (Aggregation): Hệ thống phân tích sự nhất quán=-098của các câu trả lời. Nếu đa số các bản sao bị làm nhiễu khiến mô hình từ chối thực hiện (phản hồi theo dạng kiểm duyệt an toàn), hệ thống sẽ kết luận prompt gốc là một cuộc tấn công Injection và thực hiện chặn ngay lập tức.

4. Khả năng ứng dụng thực tế và Ưu điểm vượt trội

- Tính linh hoạt (Agnostic): SmoothLLM tương thích với mọi mô hình LLM (Black-box), từ mã nguồn mở đến các API đóng, giúp doanh nghiệp dễ dàng tích hợp mà không cần can thiệp vào mã nguồn của AI.
- Hiệu suất cao: Khác với các phương pháp cần huấn luyện lại mô hình (SFT), SmoothLLM là giải pháp Training-free, giúp tiết kiệm hàng triệu USD chi phí tính toán.
- Độ tin cậy: Nghiên cứu thực nghiệm chứng minh SmoothLLM có khả năng đưa tỷ lệ tấn công thành công từ 100% xuống dưới 1%, thiết lập tiêu chuẩn bảo mật mới (State-of-the-art) cho các ứng dụng AI Agents và Chatbot thế hệ mới.

MỤC TIÊU *(Viết trong vòng 3 mục tiêu)*

1. **Vô hiệu hóa tính khả thi của các chuỗi tấn công đối kháng**

(Adversarial Robustness): Giảm tỷ lệ tấn công thành công (ASR) của các kỹ thuật Jailbreak/Injection tinh vi (như GCG, PAIR) xuống dưới mức 1%. Mục tiêu là triệt tiêu khả năng điều khiển của các chuỗi mã độc bằng cách khai thác sự mỏng manh trong cấu trúc của chúng.

2. **Duy trì độ tin cậy và hiệu năng ngôn ngữ (Utility & Consistency):** Đảm bảo hệ thống duy trì tính chính xác và chất lượng phản hồi cho người dùng hợp lệ. Mục tiêu cụ thể là giữ tỷ lệ chặn nhầm (False Positive Rate) ở mức tối thiểu và không làm suy giảm khả năng hiểu ngữ nghĩa của mô hình đối với các truy vấn bình thường.
3. **Tối ưu hóa khả năng triển khai thực tế và tính tương thích hệ thống (Scalability & Agnostic Defense):** Thiết lập một cơ chế bảo mật "Zero-shot" có khả năng bảo vệ mọi mô hình LLM (từ GPT-4 đến Llama 3) mà không cần quyền can thiệp vào tham số bên trong hay dữ liệu huấn luyện, đảm bảo tính kinh tế và khả năng mở rộng cho doanh nghiệp.

NỘI DUNG VÀ PHƯƠNG PHÁP

Nội dung 1: Cơ chế làm nhiễu cấp độ ký tự (Character-level Perturbation)

Phương pháp thực hiện:

- Sử dụng thuật toán biến đổi ngẫu nhiên trên chuỗi đầu vào P với ngân sách nhiễu q (mặc định 10%).

Thực thi thuật toán: Áp dụng 3 hàm biến đổi chính để phá vỡ cấu trúc tokenization:

- RandomSwap(P, q): Tráo đổi ngẫu nhiên vị trí các ký tự nhằm làm sai lệch các từ khóa tấn công.
- RandomPatch(P, q): Thay thế một đoạn ký tự liên tục bằng các ký tự ngẫu

nhiên từ bộ mã string.printable.

- RandomInsert(P, q): Chèn thêm ký tự lạ vào vị trí bất kỳ để làm xáo trộn cách mô hình chia tách từ (token).

Logic: Phá hủy tính tối ưu của câu lệnh đối kháng ngay từ lớp đầu vào trước khi dữ liệu được mô hình xử lý**.**

Nội dung 2: Cơ chế lấy mẫu đa tầng và bỏ phiếu đa số (Multi-sampling & Majority Voting):

Phương pháp thực hiện:

- Chuyển đổi quy trình xử lý đơn lẻ thành quy trình xử lý tập hợp biến thể $S = \{P'_1, P'_2, ..., P'_N\}$.

Thực thi thuật toán: Tạo N bản sao (ví dụ $N=10$) đã được làm nhiễu và chạy truy vấn song song qua mô hình LLM để thu được tập phản hồi $R = \{R_1, R_2, ..., R_N\}$

- Sử dụng hàm logic để phân loại: Nếu một phản hồi R_i chứa các tín hiệu từ chối (như "I'm sorry", "I cannot"), nó được gắn nhãn là *Refusal*.
- Quyết định cuối cùng: Áp dụng quy tắc bỏ phiếu đa số. Nếu số lượng phản hồi *Refusal* chiếm ưu thế, hệ thống xác định đầu vào là tấn công.

Logic: Tận dụng tính ổn định ngữ nghĩa của câu lệnh lành tính (ít bị ảnh hưởng bởi nhiễu) để đối trọng với sự mất ổn định của câu lệnh tấn công**.**

Nội dung 3: Cơ chế lớp bọc bảo vệ độc lập (Black-box Wrapper)

Phương pháp thực hiện:

- Triển khai giải pháp dưới dạng một API Gateway/Proxy nằm giữa người dùng và mô hình LLM.

Thực thi thuật toán: Input-level Transformation (Thực hiện toàn bộ logic nhiễu ký tự và quản lý đa mẫu hoàn toàn tại lớp tiền xử lý).

- Model-Agnostic Implementation: Thiết kế giải pháp chỉ làm việc trên dữ liệu văn bản thuần túy (Plaintext), không phụ thuộc vào Logits, Attention

Mask hay bất kỳ kiến trúc nội tại nào của mô hình.

Logic: Loại bỏ hoàn toàn sự phụ thuộc vào loại mô hình cụ thể, giúp doanh nghiệp cập nhật AI mới mà không cần thay đổi hay cấu hình lại lớp bảo mật, đồng thời tiết kiệm chi phí huấn luyện lại (Fine-tuning).

KẾT QUẢ MONG ĐỢI

- Giảm tỷ lệ tấn công thành công (ASR) từ mức gần 100% xuống dưới 1% đối với các cuộc tấn công dựa trên tối ưu hóa như GCG và PAIR.
- Duy trì tỷ lệ trả lời đúng cho các câu hỏi lành tính (Benign queries) ở mức trên 95% - 99%.
- Thời gian phản hồi (Latency) tăng thêm không đáng kể (trong khoảng 10-20%) khi triển khai trên hạ tầng tính toán song song.
- Khả năng bảo vệ đạt hiệu quả tương đồng trên nhiều dòng mô hình khác nhau mà không cần sửa đổi cấu trúc.

TÀI LIỆU THAM KHẢO *(Định dạng DBLP)*

[1] **Alexander Robey, Eric Wong, Hamed Hassani, George J. Pappas:** SmoothLLM: Defending Large Language Models Against Jailbreaking Attacks. *Trans. Mach. Learn. Res. (TMLR)* 2024 (2024). [Online]. Available: https://arxiv.org/abs/2310.03684

[2] **Xunguang Wang, Daoyuan Wu, Zhenlan Ji, Zongjie Li, Pingchuan Ma, Shuai Wang, Yingjiu Li, Yang Liu, Ning Liu, Juergen Rahmel:** SelfDefenD: LLMs Can Defend Themselves against Jailbreaking in a Practical Manner. Accepted by USENIX Security Symposium 2025 [Online]. Available: https://arxiv.org/abs/2406.05498

[3] **Yu Li, Han Jiang, Zhihua Wei:** DeTAM: Defending LLMs Against Jailbreak Attacks via Targeted Attention Modification (2025). [Online].

Available: https://arxiv.org/abs/2504.13562

[4] **Qiusi Zhan, Richard Fang, Henil Shalin Panchal, Daniel Kang:** Adaptive Attacks Break Defenses Against Indirect Prompt Injection Attacks on LLM Agents. [Online]. Available: https://arxiv.org/abs/2503.00061

[5]**Benji Peng, Keyu Chen, Qian Niu, Ziqian Bi, Ming Liu, Pohsun Feng, Tianyang Wang, Lawrence K.Q. Yan, Yizhu Wen, Yichao Zhang, Caitlyn Heqi Yin, Xinyuan Song:** Jailbreaking and Mitigation of Vulnerabilities in Large Language Models. [Online]. Available: https://arxiv.org/abs/2410.15236