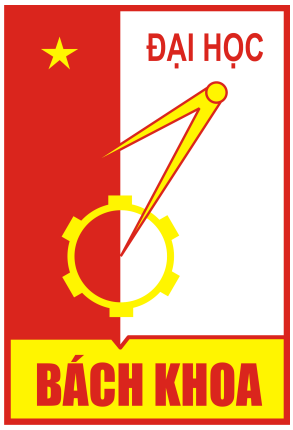
ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO

Nghiên cứu mô hình phát hiện mã độc thông qua đồ thị lưu lượng mạng

Project III

Giảng viên hướng dẫn: ThS. Bùi Trọng Tùng
Sinh viên thực hiện:

Họ và tên	*MSSV*
Đàm Ngọc Khánh	**20205207**

Mục lục

Chương 1

Giới thiệu đề tài

1.1. Đặt vấn đề

Trong thời đại ngày nay, việc tấn công mạng bằng các phần mềm độc hại (malware) đã trở thành mối đe dọa lớn đối với hệ thống thông tin và an ninh mạng. Những kẻ tấn công không ngừng tiếp tục phát triển các kỹ thuật tinh vi để tránh bị phát hiện, từ đó gây nguy hiểm và gây tổn thất lớn cho các cá nhân và tổ chức.

Một trong những thách thức lớn trong lĩnh vực an ninh mạng là khả năng phân loại lưu lượng mạng và phát hiện malware một cách hiệu quả. Hiện nay, các phương pháp truyền thống sử dụng chữ ký (signature-based) thường không đủ linh hoạt để xác định các biến thể mới và tiên tiến của malware. Do đó, việc phát triển mô hình phân loại lưu lượng mạng thông minh, có khả năng học máy (machine learning), là cực kỳ quan trọng để đối mặt với sự phức tạp ngày càng tăng của mối đe dọa mạng.

1.2. Mục tiêu và phạm vi đề tài

Mục tiêu chính của đề tài này là xây dựng và đánh giá hiệu quả của các mô hình học máy trong việc phân loại và phát hiện malware trong lưu lượng mạng. Cụ thể, với dữ liệu về lưu lượng mạng qua các thiết bị Android, em sẽ áp dụng mạng nơ-ron đồ thị (GNN - Graph Neural Network) để xây dựng một mô hình phân loại chính xác và linh hoạt.

Chương 2

Tình hình nghiên cứu hiện nay

2.1. Tổng quan

Theo một bài survey mới nhất về phát hiện malware sử dụng học biểu diễn đồ thị năm 2023[1], việc phát hiện malware phổ biến như ransomware, worms, trojan horses hoặc spyware đã trở thành một vấn đề lớn khi chúng tăng cả về số lượng và độ phức tạp. Các chương trình malware có thể xuất hiện dưới nhiều hình thức và có thể được giấu trong các chương trình đáng tin cậy khác trên các nền tảng phổ biến nhất như Android, Windows hoặc thậm chí là Web. Người dùng không ý thức thường bị đánh lừa bởi những kẻ tạo ra malware.

Đã có những nỗ lực quan trọng để ngăn chặn những mối đe dọa này. Các kỹ thuật phát hiện truyền thống chủ yếu dựa vào chữ ký và heuristics, nơi malware được phát hiện bằng cách so sánh nó với malware hiện có hoặc các mẫu độc hại đã biết. Tuy nhiên, những phương pháp này đã được biết đến vì khả năng tổng quát kém đối với các cuộc tấn công hoặc biến thể không xác định. Các phương pháp khác dựa trên hành vi xuất hiện có vẻ hiệu quả hơn bằng cách phân tích sâu hơn về malware và đánh giá hành động dự kiến của nó trước khi thực hiện nó. Tuy nhiên, những kỹ thuật như vậy có vẻ rất tốn thời gian.

Trong thập kỷ qua, Học Máy (Machine Learning - ML) và đặc biệt là Học Sâu (Deep Learning - DL) đã tạo ra một sự thay đổi lớn trong nhiều lĩnh vực, bao gồm cả an ninh mạng, bằng cách cho phép mô hình học từ dữ liệu và thích ứng với các mẫu mới. Khả năng thích ứng này làm cho những phương pháp này rất phù hợp cho nhiều nhiệm vụ, bao gồm phát hiện malware. Các nghiên cứu hiện tại về phát hiện malware sử dụng học máy chủ yếu dựa trên việc xem xét các kỹ thuật truyền thống của ML và DL được áp dụng cho dữ liệu có cấu trúc. Mặc dù đã có sự tiến bộ với các phương pháp dựa trên học này, phát hiện malware vẫn là một nhiệm vụ thách thức, vì những kẻ tạo ra malware tiếp tục phát triển các kỹ thuật phức tạp để tránh bị phát hiện.

Đối mặt với vấn đề này, học biểu diễn đồ thị (Graph Representation) đã nổi lên như một lựa chọn hứa hẹn để bắt kịp các sự phức tạp của malware. Ưu điểm của cấu trúc đồ thị so với các mô hình ML và DL truyền thống là cấu trúc đồ thị cung cấp thêm thông tin ngữ nghĩa khi có khả năng tính toán mối quan hệ không gian và kết nối giữa các thực thể.

2.2. Các loại đồ thị được sử dụng phổ biến để phát hiện malware

- **Đồ thị luồng điều khiển - Control Flow Graph:** đồ thị có hướng biểu diễn luồng điều khiển của một chương trình. Nó được sử dụng để mô hình chuỗi các lệnh được thực thi bởi một chương trình và điều kiện để chúng được thực thi.
- **Đồ thị gọi hàm - Function Call Graph:** đồ thị có hướng biểu diễn các lời gọi hàm được thực hiện bởi một chương trình. Nó được sử dụng để mô hình hóa các phụ thuộc giữa các hàm trong một chương trình và thứ tự mà chúng được gọi.
- **Đồ thị phụ thuộc chương trình - Program Dependence Graph:** đồ thị có hướng biểu diễn các phụ thuộc giữa các lệnh trong chương trình. Nó được sử dụng để mô hình các phụ thuộc dữ liệu và điều khiển giữa các lệnh trong một chương trình.
- **Đồ thị gọi hệ thống - System Call Graph:** đồ thị có hướng biểu diễn các lời gọi hệ thống được thực hiện bởi một chương trình. Nó được sử dụng để mô hình sự tương tác giữa một chương trình và hệ điều hành.
- **Đồ thị thực thể hệ thống - System Entities Graph:** đồ thị có hướng biểu diễn các thực thể trong hệ thống và các mối quan hệ giữa chúng. Nó được sử dụng để mô hình sự tương tác giữa các thực thể khác nhau trong hệ thống.
- **Đồ thị lưu lượng mạng - Network Flow Graph:** đồ thị có hướng biểu diễn luồng dữ liệu qua một mạng. Nó được sử dụng để mô hình sự tương tác giữa các nút khác nhau trong một mạng.

2.2.1 Phương pháp chung để phát hiện malware bằng học biểu diễn đồ thị

Bước đầu tiên bao gồm việc trích xuất mã nguồn từ file nhị phân, thường thông qua việc dịch mã nguồn thành ngôn ngữ hợp ngữ hoặc giải dịch thành ngôn ngữ cấp cao hơn. Trong trường hợp phát hiện malware với phân tích động, bước này giả định các đặc trưng đầu vào động như một chuỗi các lời gọi API hoặc tương tác với thực thể hệ thống. Tiếp theo, một công cụ xây dựng đồ thị được sử dụng để chuyển đổi mã nguồn thành biểu đồ, giữ nguyên ý nghĩa của chương trình. Hai bước đầu tiên này được

thực hiện bằng cách sử dụng các công cụ reverse engineering. Đồ thị có thể được tiền xử lý và được trang bị các đặc trưng được tạo bằng tay, được đặt trên các đỉnh hoặc cạnh.

Sau đó, các kỹ thuật học biểu diễn đồ thị, như GNN, sử dụng các đồ thị để học các vectơ đặc trưng, chúng bao gồm mối quan hệ và vai trò của các hướng dẫn nội bộ, chức năng hoặc lời gọi API, phụ thuộc vào biểu đồ đầu vào. Những vectơ đặc trưng này thường được tạo ra bằng cách sử dụng các biến thể GNN đã nêu ở mục trên. Một số kỹ thuật khác sử dụng các kỹ thuật nhúng từ vựng từ Ngôn ngữ Tự nhiên (NLP) để học nghĩa của mã vận hành hoặc các chức năng API, cho phép tích hợp các vectơ nhúng kết quả vào một cấu trúc đồ thị toàn cục để GNN học các thuộc tính cấu trúc. Đa số các nghiên cứu xem xét phát hiện malware như một nhiệm vụ phân loại đồ thị, trong đó vectơ đặc trưng được chuyển qua một lớp global pooling để tạo ra một vectơ đặc trưng đồ thị cố định duy nhất chứa tất cả thông tin của đồ thị. Vectơ cuối cùng sau đó có thể được phân loại bằng cách sử dụng các phương pháp Học máy truyền thống hoặc Học sâu.

2.2.2 Các phương pháp phát hiện malware bằng đồ thị trên các nền tảng

Bài survey phân tích một số phương pháp phát hiện malware bằng đồ thị đã được thực hiện với độ chính xác nhất định:

1. Phát hiện malware trên nền tảng Android:
   (a) Các loại đồ thị được dùng:
      - Đồ thị luồng điều khiển
      - Đồ thị gọi hàm
      - Đồ thị phụ thuộc chương trình
      - Đồ thị gọi hệ thống
      - Đồ thị lưu lượng mạng

   (b) Các bộ dữ liệu được dùng:
      - CICAndMal2017: bộ dữ liệu về malware dành cho Android được phát triển bởi Viện An ninh mạng Canada (CIC). Bộ dữ liệu này bao gồm 10,854 file APK được xuất bản từ năm 2015 đến 2017 trên Google Play Store. Bộ dữ liệu bao gồm 6,500 ứng dụng không độc hại và 4,354 ứng dụng malware được chia thành các lớp Benign (Không độc hại), Adware, Ransomware, SMS và Riskware. Đối với mỗi tình huống, các gói tin mạng cũng được thu thập và chuyển đổi thành các lưu lượng sử dụng CICFlowMeter. Công cụ này tạo ra 80 đặc trưng cho mỗi lưu lượng, dựa trên thống kê từ các gói tin chứa trong lưu lượng đó.

- CICMalDroid: cũng được công bố bởi Viện An ninh mạng Canada. Bao gồm các file APK và các đặc trưng được trích xuất ra các file CSV.
- AndroZoo: Các ứng dụng Android được tổng hợp file APK bởi Đại học Luxembourg.
- Drebin: được cung cấp thông qua dự án MobileSandbox, bộ dữ liệu này cũng cung cấp các ứng dụng Android malware. Mỗi file APK được trích xuất 10 đặc trưng và tất cả các file đều là malware.
- MalNet: bộ dữ liệu lớn chứa các Function Call Graphs (FCGs) được trích xuất từ các file APK của AndroZoo.

2. Phát hiện malware trên nền tảng Windows:
   (a) Các loại đồ thị được dùng:
   - Đồ thị luồng điều khiển
   - Đồ thị gọi hàm
   - Đồ thị thực thể hệ thống

   (b) Các bộ dữ liệu được dùng:
   - Microsoft Malware Classification Challenge (MMCC): bộ dữ liệu chứa hơn 20000 mẫu malware. Đối với mỗi file nhị phân, bộ dữ liệu cung cấp hai biểu diễn dữ liệu: bytecode và mã disassembly (được giải dịch bằng IDA Pro). Mã assembly sau đó có thể được sử dụng để xây dựng đồ thị gọi hàm.
   - VirusShare, VirusTotal: hai trang web cung cấp các ứng dụng malware trên windows.

3. Phát hiện malware trên Web:
   (a) Các hướng tiếp cận:
   - Tiếp cận theo mã nguồn
   - Tiếp cận theo hệ thống phân giải tên miền
   - Đồ thị lưu lượng mạng

2.2.3 Vấn đề về tấn công đối kháng

Tấn công đối kháng là quá trình thay đổi những đối tượng hoặc dữ liệu đầu vào của một mô hình máy học để tạo ra một kết quả không mong muốn hoặc gây nhầm lẫn cho mô hình. Mục tiêu của việc này là làm cho mô hình đưa ra dự đoán sai lầm hoặc thay đổi kết quả mong đợi.

Trong nghiên cứu về phát hiện mã độc dựa trên đồ thị, các tấn công đối kháng đặc biệt được thiết kế cho hệ thống phân loại GNN. Những tấn công này thường sửa đổi đặc trưng của nút và cạnh hoặc tương tác

trực tiếp với cấu trúc đồ thị bằng cách thêm hoặc xóa nút và cạnh. Trong trường hợp của mã độc, việc xóa nút hoặc cạnh không phải là phương pháp hiệu quả vì nó có thể làm mất chức năng của malware. Một số phương pháp tấn công đã được đề cập, bao gồm sự chèn nút mạnh nhất, sử dụng hướng dẫn từ đạo hàm, và sử dụng học tăng cường để thực hiện các sửa đổi đồ thị. Các tấn công này đã đạt được kết quả ấn tượng, với tỷ lệ phân loại sai cao, đặt ra thách thức về tính chống lại của các phương pháp phát hiện dựa trên GNN.

2.3. Các thách thức hiện nay

Học biểu diễn đồ thị chỉ mới được áp dụng vào việc phát hiện malware trong những năm gần đây. Vì vậy, vẫn còn nhiều thách thức để đạt được các mô hình mạnh mẽ và chính xác. Một số thách thức bao gồm:

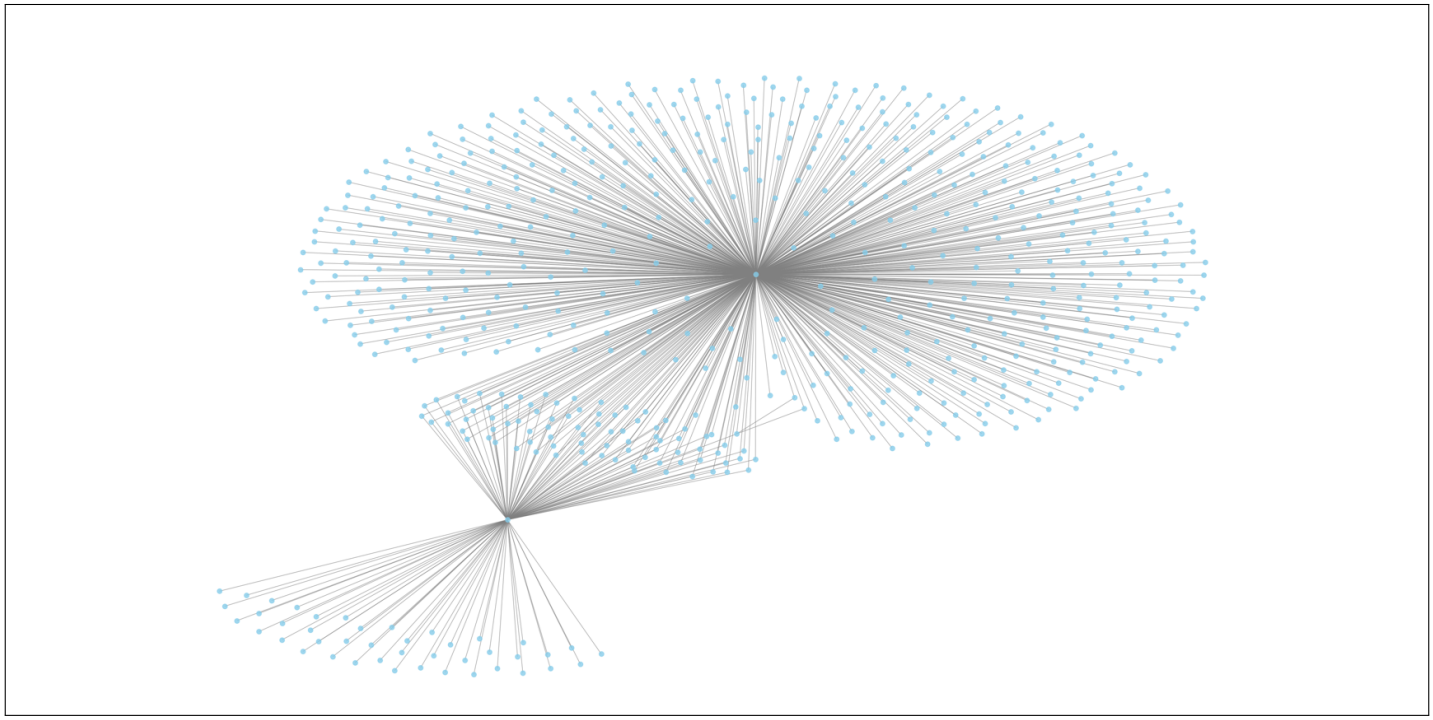
- Xây dựng một bộ dữ liệu tiêu chuẩn và đa dạng: các bài báo thường được đánh giá trên các ví dụ khác nhau, điều này làm cho so sánh giữa các nghiên cứu trở nên không hiệu quả. Sự tạo ra một bộ dữ liệu cơ sở lớn và đa dạng có thể giúp so sánh các mô hình một cách hiệu quả.
- Nghiên cứu về tấn công đối kháng: mô hình cần có khả năng chống lại các tấn công đối kháng, vì các tấn công này đang trở nên ngày càng phổ biến và có thể đe dọa tính độc lập của mô hình.
- Nâng cao khả năng giải thích được của mô hình: phát triển các phương pháp giải thích để giúp hiểu rõ hơn về quyết định của mô hình. Điều này làm tăng tính minh bạch và sự tin cậy của mô hình, đặc biệt là trong lĩnh vực an ninh mạng.

Chương 3

Xây dựng mô hình

3.1. Dataset

Tập dữ liệu dùng để xây dựng các đồ thị được trích xuất từ dữ liệu lưu lượng trong bộ dữ liệu CICAndMal2017. Lý do được sử dụng bởi bộ dữ liệu nàt bởi nó cung cấp đủ số lượng mẫu từ các phần mềm độc hại đa dạng các loại và các mẫu lành tính. Các ứng dụng được thực thi trên một thiết bị vật lý thực tế thay vì mô phỏng hoặc trên máy ảo giúp nắm bắt chính xác hình vi hơn.
Tập dữ liệu được trích xuất bao gồm 2126 mẫu, trong đó mỗi mẫu tương ứng với một phiên bản của ứng dụng Android được cài đặt và được thực hiện trên điện thoại di động. Đối với mỗi mẫu, tất cả các luồng mạng trong mạng trong quá trình thực thi sẽ được ghi lại. Đối với mỗi luồng, 80 đặc trưng được ghi lại, bao gồm số lượng gói được gửi, độ lệch trung bình và tiêu chuẩn của độ dài gói và tối thiểu và thời gian tương tác tối đa của các gói.
Đối với mỗi mẫu, có sẵn 3 nhãn khác nhau, nhưng trong nghiên cứu này chỉ sử dụng nhãn nhị phân là có bị tấn công hay không.



Hình 3.1: Đồ thị lưu lượng mang được trích xuất bởi bộ dữ liệu

3.2. Xử lý dữ liệu

3.2.1 Tiền xử lý dữ liệu

Từ bộ dữ liệu, ta có được 1 list các đồ thị lưu lượng mạng với thuộc tính cạnh được mô tả bằng danh sách các ma trận thuộc tính cạnh tương ứng với kích thước m x d với m là số cạnh của đồ thị và d số thuộc tính
Ngoài ra, bởi đây là nghiên cứu học có giám sát nên mỗi đồ thị đều đã được dán nhãn malware hoặc benign

3.2.2 Học biểu diễn đồ thị

Đầu vào của mô hình là đồ thị có hướng G=(V, E) với ma trận A kích thước n x n chỉ ra quan hệ giữa các đỉnh và ma trận X kích thước m x d với n = |V|, m = |E|.
Các biểu diễn tiềm ẩn của cạnh và nút được tính toán và cuối cùng trích xuất ra 1 vector h cho mỗi nút trong đồ thị. Một vector mô tả 1 cách trực quan cách 2 điểm ip tương tác với nhau, trong bài báo đã tham khảo, các bước biến đổi và lan truyền được tính toán tuần tự như sau:

$$E_0 = f_1(X) : m * h$$

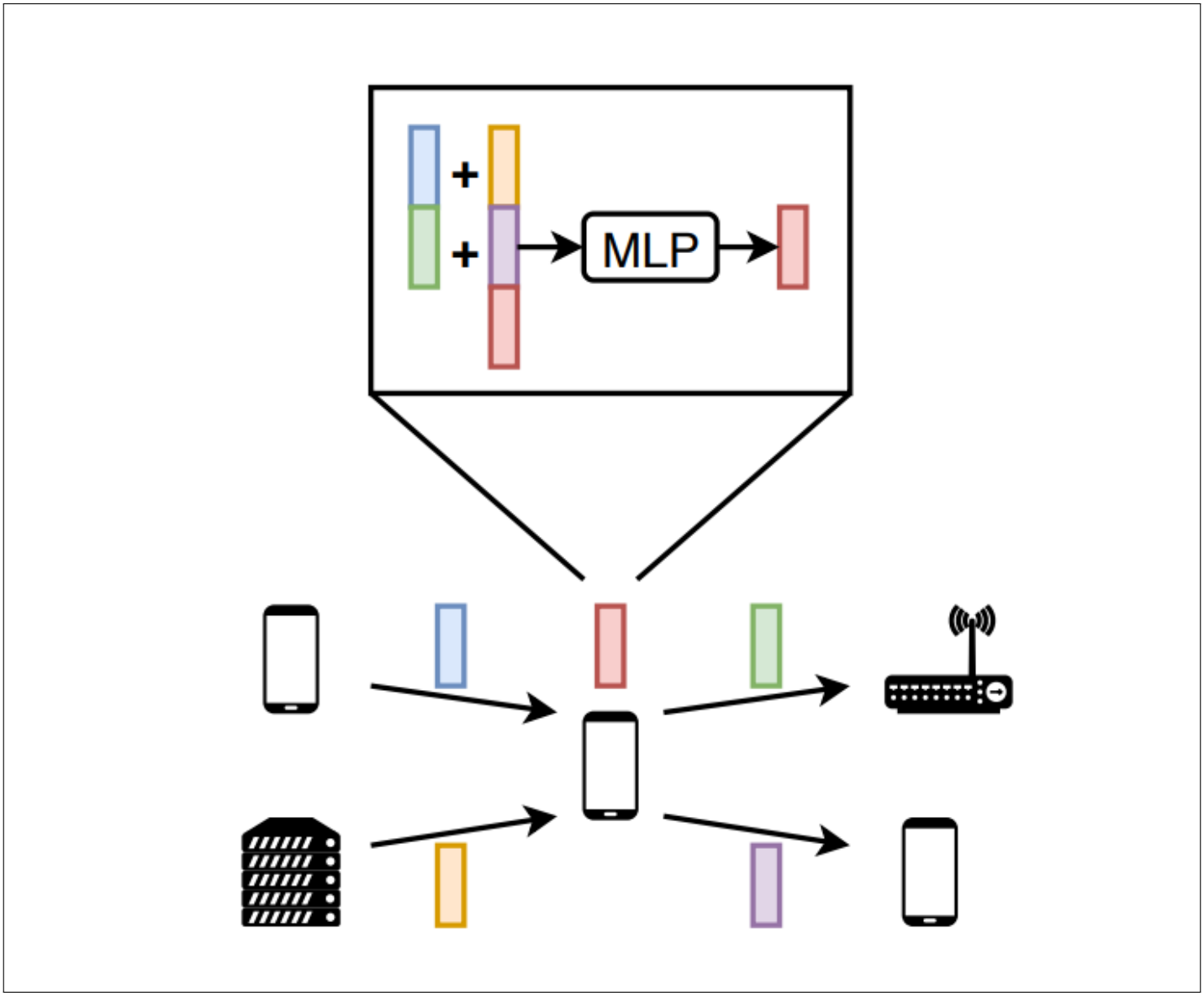
$$H_0 = f_2([B_{in}E_0, B_{out}E_0]) : n * h$$

$E_1 = f_3([B_{in}^T H_0, B_{out}^T H_0, E]) : m * h$

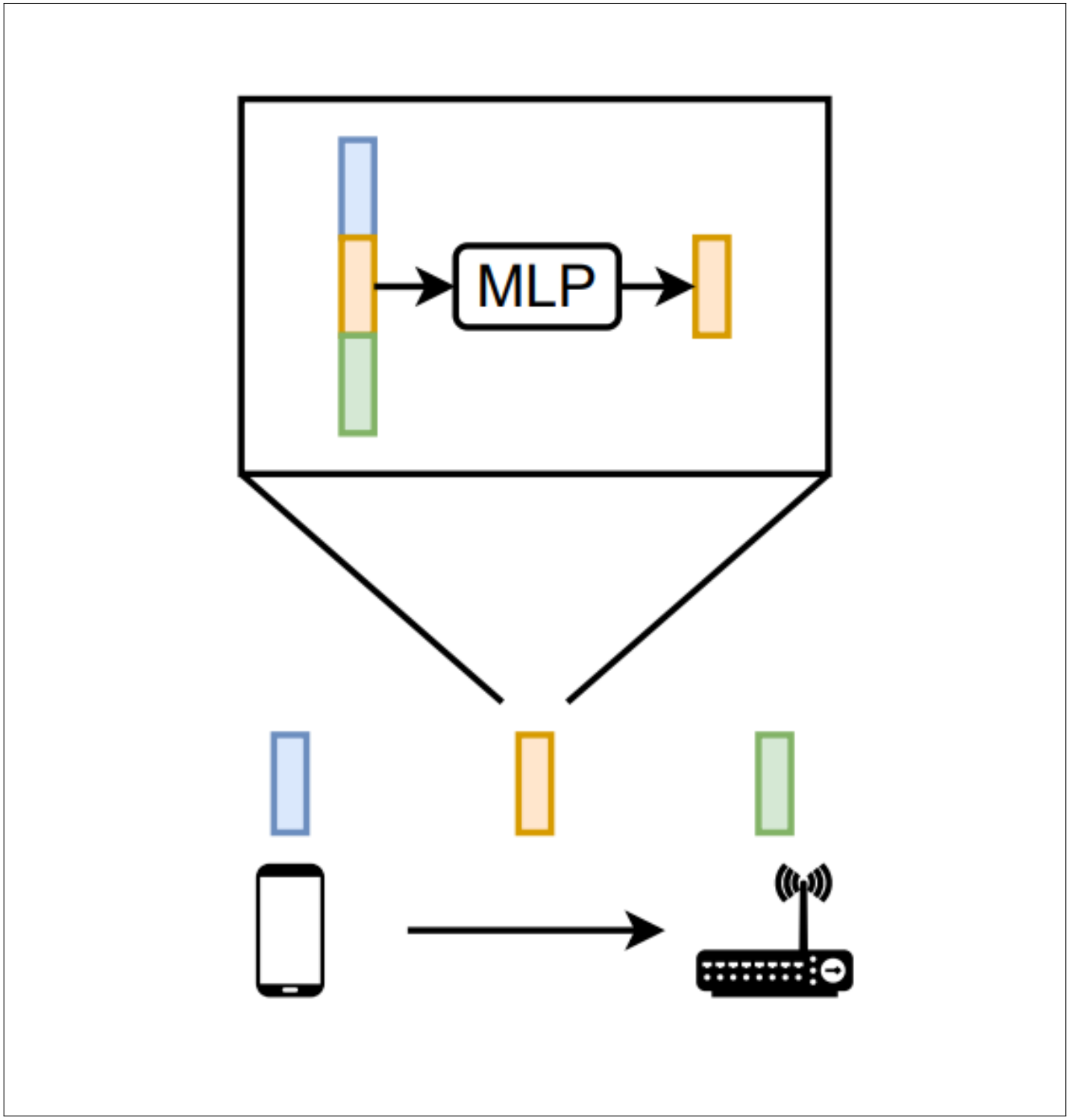
$H_1 = f_4([B_{in} E_1, B_{out} E_1, H_0])$

f_1, f_2, f_3, f_4 là lớp MLP: $f_i(X) = q(XW_i + b_i)$

Ma trận B_{in}, B_{out} kích thước n * m với :
$(B_{in})_{ij} = 1$ nếu tồn tại đỉnh vk thuộc $V : e_j = (v_k, v_i)$ tức vị trí hàng i cột j sẽ = 1 nếu tồn tại 1 cạnh đi vào đỉnh v_i, = 0 với trường hợp còn lại. Với B_{out} thì = 1 nếu tồn tại 1 cạnh đi ra từ đỉnh v_i và ngược lại sẽ bằng 0.
Lớp đầu tiên sẽ tìm ra cách các điểm đầu cuối tương tác với nhau bằng cách biến đổi từ vector đặc trưng cạnh (công thức 1) và sau đó tính toán biểu diễn nút bằng các tổng hợp các vector đặc trưng từ cạnh lân cận (công thức 2).
Lớp thứ 2, tìm hiểu các điểm giao tiếp gián tiếp với 2-hop hàng xóm. Bước đầu tiên các thuộc tính cạnh được cập nhật lại bằng nối ma trận theo hàng ngang như công thức 3. Cuối cùng thu được vector ẩn h.



Hình 3.2: Minh họa cập nhật thuộc tính đỉnh



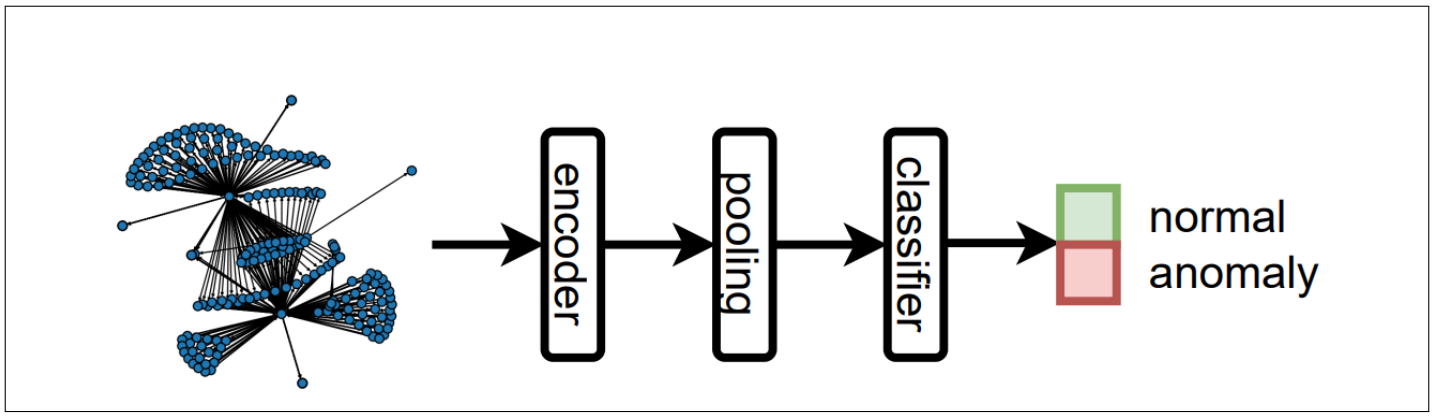
Hình 3.3: Minh họa cập nhật thuộc tính cạnh

3.3. Mô hình phân loại

Trong phạm vi nghiên cứu này chỉ phân loại nhị phân có giám sát thì theo bài báo tham khảo, người ta đã gắn thêm 2 là pooling để kết tập tất cả

các vector đặc trưng nút và tầng thứ 2 để dự đoán nhãn từ đồ thị đã pool.

h = pool(H_1)
y = softmax(Wh + b)
Ở trên, pool biểu thị một hàm gộp, chẳng hạn như giá trị trung bình theo phần tử hoặc tối đa, tổng hợp tất cả các biểu diễn nút thành một vectơ nhúng duy nhất cho toàn bộ biểu đồ. Dự đoán được tính toán bởi một lớp dự đoán dày đặc với các tham số có thể học được và sau đó kích hoạt softmax. Sau các lần huấn luyện sẽ tối ưu hóa mất mát cross-entropy.



Hình 3.4: Minh họa các layer trong model

Những tìm hiểu trên được em tìm hiểu dựa trên bài báo [2], nên có thể còn có sai sót

Chương 4

Hướng phát triển

Trong tương lai, cần thực hiện nghiên cứu chuyên sâu về độ chính xác và độ bền của các mô hình GNN trong môi trường thực tế. Điều này bao gồm việc đối mặt với các thách thức như sự đa dạng và độ biến đổi của malware để đảm bảo mô hình là đủ chính xác và linh hoạt. Đồng thời, mô hình phát hiện malware bằng GNN có thể được ứng dụng để triển khai một hệ thống học không giám sát phát hiện và phân loại malware qua lưu lượng mạng theo thời gian thực.

Chương 5

Tài liệu tham khảo

[1] Tristan Bilot, Nour El Madhoun, Khaldoun Al Agha, and Anis Zouaoui. A survey on malware detection with graph representation learning. 2023.

[2] Julian Busch, Volker Tresp, Anton Kocheturov, and Thomas Seidl. Nf-gnn: Network flow graph neural networks for malware detection and classification. *33rd International Conference on Scientific and Statistical Database Management (SSDBM 2021), July 6–7, 2021, Tampa, FL, USA*, 2021.