TẠP CHÍ KHOA HỌC ĐẠI HỌC HẠ LONG
Scientific Journal of Ha Long University
http://uhl.edu.vn/

KỸ THUẬT HỌC SÂU TRONG PHÁT HIỆN TẤN CÔNG BOTNET

Đoàn Trung Sơn[1*], Nguyễn Thị Khánh Trâm[1], Phạm Minh Hiếu[2]

[1]Khoa Công nghệ thông tin, Trường Đại học Phenikaa
[2]Phòng PA05, Công an tỉnh Hà Tĩnh

** Email: son.doantrung@gmail.com*

Ngày nhận bài:15/6/2022 *Ngày nhận bài sửa sau phản biện: 20/10/2022* *Ngày chấp nhận đăng: 24/10/2022*

TÓM TẮT

Trong quá trình phát triển kinh tế – xã hội của đất nước hiện nay thì chuyển đổi số là một yêu cầu tất yếu. Một trong những rào cản thúc đẩy chuyển đổi số thành công là các vấn đề liên quan đến an toàn thông tin và an ninh mạng. Thực tế, Việt Nam nằm trong số các quốc gia bị tấn công mạng nhiều nhất trong khu vực nhưng lại có chỉ số an ninh mạng thấp nhất. Thời gian qua, thế giới chứng kiến sự bùng nổ một cách mạnh mẽ chưa từng có của Deep learning. Bên cạnh sự phát triển của Công nghệ thông tin, các mối đe doạ về an ninh, an toàn cũng ngày càng tăng lên mà một trong số đó chính là mạng Botnet. Mạng Botnet ngày càng phức tạp và khó phát hiện, các kỹ thuật truyền thống không còn phát huy được nhiều tác dụng, vì vậy một trong những vấn đề cấp thiết hiện nay là tìm ra được giải pháp thật hiệu quả trong phát hiện mạng Botnet. Dựa trên những đặc điểm ưu thế của học sâu như khả năng mở rộng, hiệu suất và thời gian thực hiện, khả năng diễn giải, v.v., nhóm tác giả đã tiến hành cài đặt và đánh giá ba phương pháp phát hiện Botnet dựa trên học sâu, kết quả thu được là vượt trội. Do đó, trong bài báo này, nhóm tác giả đã sử dụng kỹ thuật học sâu, đề xuất xây dựng một mô hình mạng nơron ba lớp trong việc phát hiện, cảnh báo các cuộc tấn công sử dụng mạng Botnet. Qua so sánh và đánh giá, kết quả đạt được của mô hình mạng nơron được đề xuất là tốt hơn so với các phương pháp khác như SVM, RNN, LSTM.

Từ khóa: *Botnet, BoTShark-CNN, BoTShark-SA, học sâu.*

DEEP LEARNING TECHNIQUES TO DETECT BOTNET

ABSTRACT

During the economic and social development of the country, digital transformation is an indispensable requirement. Information and network security are two major impediments to successful digital transformation. Cyberattacks are happening every day, every hour and in fact, Vietnam is among the countries most attacked by cyberattacks in the region but has the lowest cybersecurity index. In the past time, the world has witnessed an unprecedented explosion of deep learning. Besides the development of information technology, security and safety threats are also increasing, one of which is the Botnet network. Because Botnet networks are becoming more complex and difficult to detect, traditional techniques are no longer effective, and one of the most pressing issues today is to find an effective solution for detecting botnets. Based on the advantages of deep learning such as scalability, performance and execution time, interpretability, etc., the authors have installed and evaluated 3 methods of Botnet detection. The results obtained are outstanding. Therefore, in this paper, the authors have used the deep learning technique and proposed to build a 3-layer neural network model for detecting and warning against attacks using Botnets. Through comparison and evaluation, the results obtained by the proposed neural network model were better than other methods such as SVM, RNN, and LSTM.

Keywords: *Botnet, BoTShark-CNN, BoTShark-SA, deep learning.*

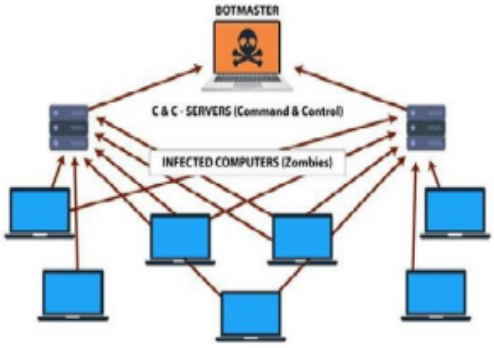
1. ĐẶT VẤN ĐỀ

Botnet là thuật ngữ viết ngắn của "Bots network", chỉ mạng lưới các máy tính nhiễm mã độc (Bots/Zombie) và bị chi phối bởi một máy tính khác. Mạng lưới Botnet càng lớn thì độ nguy hiểm càng cao. Thực tế, Botnet là một nhóm các thiết bị internet bị xâm phạm, chúng được điều khiển từ xa bởi các tội phạm mạng. Tội phạm mạng sử dụng Botnet để khởi động các cuộc tấn công phối hợp và thực hiện các hoạt động độc hại khác. Từ "Botnet" là sự kết hợp của hai từ, "Robot" và "Network". Ở đây, một tên tội phạm mạng thực hiện vai trò của một "botmaster" sử dụng virus Trojan để xâm phạm bảo mật của một số máy tính và kết nối chúng vào mạng vì mục đích xấu. Mỗi máy tính trên mạng hoạt động như một "Bot" và được kẻ xấu kiểm soát để lây truyền Malware, Spam hoặc nội dung độc hại nhằm khởi động cuộc tấn công. Số lượng, quy mô, mức độ nguy hiểm và đặc biệt là khả năng ẩn dấu của mạng Botnet ngày càng tinh vi, phức tạp. Tại Việt Nam, theo thông tin từ Trung tâm ứng cứu khẩn cấp máy tính Việt Nam (VNCERT), trong hai quý đầu năm 2019, mỗi ngày có gần 100.000 địa chỉ mạng (IP) của Việt Nam truy vấn, kết nối đến các mạng lưới máy tính (Botnet) và có tới 6219 sự cố tấn công mạng vào các website của Việt Nam.

BotHunter là một trong những hệ thống phát hiện Botnet dựa trên hành vi ra đời sớm nhất, nó hoạt dộng dựa vào việc sử dụng phần mềm SNORT để tạo ra các cảnh báo về hành vi của từng máy riêng biệt. Tuy nhiên, việc nó hoạt động bằng cách quan sát các tải trọng gói khiến cho hệ thống này không có nhiều tác dụng trong việc phát hiện các mạng Botnet đã mã hóa kết nối. Vì vậy, BotMiner xuất hiện, BotMiner hoạt động bằng cách nhóm hành vi của các máy khác nhau trong cùng một mạng Botnet lại.

Mặt khác, trong thời gian đầu này đã có một số nghiên cứu về cách phát hiện các luồng mạng độc hại của Botnet dựa trên học máy, nhưng các luồng mạng này chỉ là từ các máy nội bộ trong mạng LAN. Sau đó, học máy đã trở thành một kỹ thuật phổ biến trong phát hiện Botnet. Hệ thống hai giai đoạn bao gồm giai đoạn trích xuất đặc tính và giai đoạn học máy. Hệ thống này được sử dụng để phát hiện các mạng Botnet có kiến trúc dựa trên IRC (Internet Relay Chat – một giao thức được thiết kế cho hoạt động liên lạc theo kiểu hình thức tán gẫu thời gian thực dựa trên kiến trúc client-server), nó hoạt động bằng cách sử dụng phương pháp phân loại Bayesian để phát hiện lưu lượng mạng từ máy C&C (Command-and-Control) của Botnet, hệ thống này cho kết quả là 90% tỷ lệ phát hiện và 15.4% tỷ lệ phát hiện sai (tỷ lệ phát hiện sai vẫn còn cao) (Abu Rajab và cs., 2006).

Một số cách phát hiện Botnet khác hoạt động dựa trên truy vấn DNS, và nó đã đạt được tỷ lệ phát hiện truy vấn DNS độc hại lên tới 92.5 % (Trần Thị Hằng, 2020). Tuy nhiên, việc phát hiện dựa vào các truy vấn DNS khiến cho các hệ thống chỉ có thể hoạt động được với các mạng Botnet sử dụng hệ thống DNS để tìm kiếm các máy chủ C&C của chúng (chủ yếu các mạng này là mạng Botnet tập trung). Hệ thống dùng để phát hiện các luồng mạng Botnet P2P hoạt động bằng cách giả định luồng mạng được tạo ra bởi người dùng sẽ giao động mạnh không giống như luồng mạng của Botnet P2P. Hệ thống này đạt được tỷ lệ phát hiện lên tới 98%, nhưng tỷ lệ phát hiện sai lại cao (30%) (Trần Thị Hằng, 2020).



Hình 1. Cấu trúc điển hình của mạng Botnet

Botnet lây lan rất nhanh, cùng với đó, các loại Botnet ngày càng tiến hóa và phát triển, càng nguy hiểm và khó phát hiện. Do khả năng lây lan dễ dàng của nó, bất kỳ ngành, nghề, lĩnh vực nào cũng đều đứng trước nguy cơ bị xâm hại bởi Botnet. Ngày nay, môi

trường mạng ngày càng trở nên phức tạp, vì vậy, yêu cầu về bảo mật càng khó khăn hơn bao giờ hết. Với việc Botnet được điều khiển bởi botmaster thông qua kênh C&C, ngoài việc dễ dàng che dấu vết nó còn khiến cho Botnet trở thành một công cụ hữu hiệu cho tội phạm mạng thực hiện nhiều loại hành vi nguy hiểm khác nhau. Một số hành vi Botnet có thể thực hiện đó là: gửi tin nhắn rác, lừa đảo, gian lận nhấp chuột, tấn công từ chối dịch vụ phân tán (Distributed Denial of Service – DDoS) và phát tán chương trình độc hại. Các Botnet được sử dụng thường xuyên trong các cuộc tấn công DDoS. Một kẻ tấn công có thể điều khiển số lượng lớn máy tính bị chiếm quyền điều khiển tại một trạm từ xa, khai thác băng thông của chúng và gửi yêu cầu kết nối tới máy đích.

2. CÁC KĨ THUẬT PHÁT HIỆN BOTNET VÀ KĨ THUẬT HỌC SÂU

2.1. Một số kĩ thuật phát hiện Botnet

Haddadi & Zincir-Heywood (2017) đã lựa chọn đặc tính để phát hiện sự bất thường. Phương pháp phát hiện Botnet sớm thông qua hai giai đoạn do Wang & Paschalidis (2017) thực hiện. Stevanovic & Pedersen (2014) đã giới thiệu một phương pháp phát hiện Botnet dựa trên luồng dữ liệu. Thuật toán rừng ngẫu nhiên đạt độ chính xác 94%. Kirubavathi & Anitha (2016) đã giới thiệu một phương pháp phát hiện Botnet dựa trên mô hình hóa hành vi của lưu lượng mạng bằng cách sử dụng các đặc tính của luồng dữ liệu và phương pháp machine learning giám sát (supervised machine learning). Guntuku và cs. (2013) đã tích hợp một mô hình Bayesian chính quy để tiền xử lý các đặc tính và chọn bộ các đặc tính đại diện nhất. Mặc dù kích thước của tập dữ liệu lưu lượng mạng lành tính và các danh mục của chúng không được đưa ra, nhưng phương pháp được đề xuất vẫn phát hiện Botnet với độ chính xác 99.2%.

Mạng Botnet ngày một phức tạp và khó phát hiện, các kỹ thuật truyền thống không còn phát huy được nhiều tác dụng, vì vậy một trong những vấn đề cấp thiết hiện nay là tìm ra được một giải pháp thật hiệu quả trong phát hiện mạng Botnet. Deep learning (học sâu) nổi lên như là một giải pháp mới đầy tiềm năng trong phát hiện Botnet (Hoàng Xuân Dậu và cs., 2019).

Tác giả đã đánh giá ba phương pháp phát hiện Botnet dựa trên học sâu đó là: BoTShark-SA (Botnet Traffic Shark – Stacked Autoencoders), BoTShark-CNN (Botnet Traffic Shark – Convolutional Neural Networks) và phương pháp phát hiện Botnet dựa trên luồng dữ liệu lạ thường. Kết quả cho thấy rằng BoTShark-SA và BoTShark-CNN đều đạt được tỉ lệ giá trị dương tính dự đoán chính xác trên giá trị dương tính thực sự TPR (True Positive Rate) lớn hơn hoặc bằng 0.91 trên bộ dữ liệu ISCX. BoTShark hoạt động trên tất cả các đặc tính cơ bản và không có đặc tính nào được lọc bởi các chuyên gia. BoTShark-SA đạt TPR là 0.91 và tỉ lệ dương tính sai FPR (False Positive Rate) là 0.15, còn BoTShark-CNN đạt tỷ lệ TPR cao hơn 0.92. Qua so sánh cho thấy phương pháp tiếp cận dựa trên nền tảng học sâu được đề xuất đạt được độ chính xác phát hiện Botnet cao hơn với tỷ lệ dương tính giả rất thấp.

Kết quả đánh giá ba phương pháp trên cho thấy các phương pháp này vượt trội hơn so với các phương pháp khác có liên quan, có thể phát hiện ra các Botnet đạt độ chính xác đạt tới 99%. Hơn nữa, thời gian huấn luyện mô hình là chấp nhận được khi áp dụng lên một bộ Botnet khá lớn, kết quả phát hiện rất chính xác và rất có tiềm năng ứng dụng vào thực tế. Bên cạnh đó, mô hình phát hiện Botnet được xây dựng dựa trên mạng nơron nhân tạo (Artificial Neural Network – ANN) cùng với đầu ra là hàm kích hoạt đầu ra Softmax cũng là mô hình được áp dụng chủ yếu và hiệu quả nhất.

2.2. Kĩ thuật học sâu phát hiện mạng Botnet

Qua thực nghiệm với ba phương pháp phát hiện Botnet dựa trên học sâu là BoTShark-SA, BoTShark-CNN và phương pháp phát hiện Botnet dựa trên luồng dữ liệu lạ thường, có thể thấy kết quả đạt được là vượt trội. Do đó, nhóm tác giả đề xuất sử dụng kỹ thuật học sâu để phát hiện mạng Botnet. Trong bài báo này, nhóm tác giả đã tiến hành chạy thực

nghiệm và đề xuất một mô hình mạng nơron dựa trên học sâu, sử dụng bộ dữ liệu Bot-IoT.

Mô hình mạng nơron được xây dựng dựa trên hàm kích hoạt là hàm Softmax. Mô hình được huấn luyện và kiểm tra với tập train và test có 10 đặc tính tốt nhất đã được trích xuất trước đó. Mô hình được xây dựng trên nền tảng là Keras với TensorFlow hỗ trợ. Ứng dụng được sử dụng đó là ứng dụng Google Collaboration của Google.

Giới thiệu bộ dữ liệu: Bộ dữ liệu được tác giả sử dụng là Bot-IoT được xây dựng bởi Nickolaos Koroniotis, Nour Moustafaa, Elena Sitnikova, Benjamin Turnbull (Koroniotis và cs., 2019) đến từ trường Đại học New South Wales, Canberra, Áo. Bộ dữ liệu là sự kết hợp của các luồng mạng IoT giả lập thường cùng với các phương pháp tấn công đa dạng. Bộ dữ liệu rất lớn với 72000000 bản lưu (record), với 16.7 GB đối với file CSV và 69.3 GB đối với tệp Pcap.

Lý do nhóm tác giả áp dụng bộ dữ liệu này là bởi vì hiện nay, mục tiêu tấn công chủ yếu của các mạng Botnet là các thiết bị IoT, đặc biệt là các thiệt bị rẻ tiền, kém chất lượng và bảo mật thấp. Trong khi đó, các thiết bị IoT đang phát triển một cách mạnh mẽ và nhanh chóng với lưu lượng dữ liệu khổng lồ. Các hệ thống IoT nổi tiếng hiện nay có thể kể đến như là: Smart city, Smart healcare, Smart home và các thiết bị IoT công nghiệp. Và Botnet, dưới sự điều khiển của BotMaster thông qua máy chủ C&C, đã tiến hành rất nhiều các cuộc tấn công vào các thiết bị IoT, trong đó các hình thức tấn công chủ yếu là: DDos, Keylogging, Phishing, Spamming, Click fraud, Identity theft và kể cả đó là phát tán các mã độc Botnet.

Bộ dữ liệu Bot-IoT là một bộ dữ liệu đã được xây dựng và phân tích một cách triệt để và đầy đủ, để thông qua đó có thể áp dụng một cách dễ dàng các phương pháp học máy, học sâu vào bộ dữ liệu. Bộ dữ liệu này đã thể hiện được ưu điểm của nó khi đem so sánh với các bộ dữ liệu khác như sau:

Bảng 1. So sánh bộ dữ liệu Bot-IoT với các bộ dữ liệu khác

Bộ dữ liệu	Cài đặt thử nghiệm thực tế	Lưu lượng thực tế	Dữ liệu được gán nhãn	Bắt đầy đủ gói tin	Trích xuất các đặc tính mới
Darpa98	T	F	T	T	F
KDD99	T	F	T	T	T
DEFCON-8	F	F	F	T	F
UNIBS	T	T	T	T	F
CAIDA	T	T	F	F	F
LBNL	F	T	F	F	F
UNSW-NB15	T	T	T	T	T
ISCX	T	T	T	T	T
CICIDS 2017	T	T	T	T	T
TUIDS	T	T	T	T	T
Bot-IoT	T	T	T	T	T

Trong đó, các chỉ số T, F được hiểu là True (T) – đúng/ có và False (F) – sai/ không. Ví dụ, bộ dữ liệu Darpa98 có chỉ số cài đặt thử nghiệm thực tế là T được hiểu là có thể cài đặt thử nghiệm thực tế. Với chỉ số trích xuất đặc tính mới là F được hiểu là bộ dữ liệu Darpa98 không trích xuất các đặc tính mới.

Qua so sánh, bộ dữ liệu Bot-IoT thể hiện sự tốt hơn (T) với các chỉ số đánh giá: Cài đặt thử nghiệm thực tế, Lưu lượng thực tế, Dữ liệu được gán nhãn, Bắt đầy đủ gói tin, Trích xuất các đặc tính mới. Đây là những chỉ số đặc trưng trong việc phát hiện mạng Botnet.

Bộ dữ liệu được xây dựng dựa trên ba thành phần đó là: nền tảng mạng, mô phỏng các thiết bị IoT và cuối cùng là trích xuất đặc tính và điều tra, phân tích.

Sau khi thu thập được bộ dữ liệu Bot-IoT, nhóm tác giả tiến hành giai đoạn tiền xử lý dữ liệu để đưa vào huấn luyện cho mô hình.

Tiền xử lý dữ liệu: Trước tiên, cần phải nhập dataset vào. Đầu tiên, chúng ta khởi tạo đường dẫn tới dataset, sau đó sử dụng hàm đọc dữ liệu có định dạng CSV của Python.

Tiếp theo, chúng ta sẽ lấy các đặc trưng của dữ liệu, ở đây các cột đặc tính dùng để định danh luồng dữ liệu như saddr, daddr, proto, sport, dport sẽ được loại bỏ. Ngoài ra, chúng ta cũng loại bỏ các cột nhãn trong dữ liệu như attack, category và subcategory. Đây là các nhãn của các luồng dữ liệu nên chúng ta tách riêng.

Ngoài ra, nhãn được dùng để phân biệt là category. Chúng ta cần chuyển các nhãn của cột category thành các giá trị số. Nó tương tự như từ điển, gồm năm giá trị: {0:A, 1:B, 2:C, 3:D, 4:E}. Chỗ nào nhãn A nó thay bằng số 0, nhãn B nó thay số 1. Sau đó chúng ta cần chuyển các con số tương ứng với nhãn thành các vector. Ví dụ, nhãn cột đầu tiên là 0 thì chuyển thành vector năm chiều là [1, 0, 0, 0, 0], nhãn cột thứ hai là 4 thì sẽ chuyển thành [0, 0, 0, 1, 0]. Bởi vì mô hình mạng nơron yêu cầu đầu vào của nhãn dưới dạng một vector nên chúng ta cần phải chuyển nhãn của nó thành vector.

Đối với bộ dữ liệu đã trích xuất, chúng ta tiến hành phân loại các cột trong bộ dữ liệu thành các cột dạng chữ và dạng số:

```
all_keys = X_train.keys()

num_keys = X_train.iloc[:, :-
2].select_dtypes(exclude=['object'])
.keys()

cat_keys = X_train.iloc[:, :-
2].select_dtypes(include=['object'])
.keys()print(all_keys)
```

Trong đó, biến num_keys chứa các cột giá trị số và biến cat_keys chứa các cột giá trị chữ, lệnh iloc dùng để duyệt từng hàng trong cột (keys) còn lệnh select_dtypes dùng để xác định kiểu dữ liệu cần chọn. Đối với các cột có giá trị là số, chúng ta cần phải tiến hành điều chỉnh dữ liệu của các đặc tính về chung một tỉ lệ với khoảng giá trị đủ nhỏ, mục đích của nó là giúp cho các trình phân loại hoạt động chính xác và hiệu quả nhất. Kỹ thuật được áp dụng ở đây là MinMax Scaling

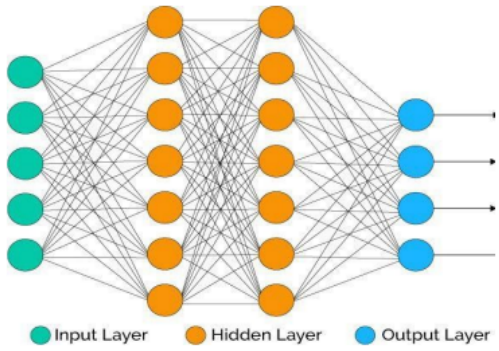
$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}.$$

Công thức trên sẽ chuẩn hóa các đặc tính về giá trị trong khoảng [0, 1]. Tuy nhiên, do các giá trị trong bộ dữ liệu quá lớn nên không thể cứ thế áp dụng MinMax Scaling nên trước tiên cần phải sử dụng công thức logarit cơ số 10 để giảm kích thước dữ liệu.

Tương tự đối với các cột có giá trị là chữ, chúng ta cũng sẽ sử dụng MinMax Scaling để đưa dữ liệu về trong khoảng [0, 1]. Tuy nhiên, trước đó, chúng ta cũng cần phải có một bước chuyển các giá trị dạng chữ về dạng số theo phương pháp từ điển.

Sau khi đã xây dựng và tiền xử lý bộ dữ liệu, nhóm tác giả tiến hành xây dựng mô hình mạng nơron.

Xây dựng mô hình mạng nơron: Mô hình mạng nơron gồm ba lớp như sau:



Hình 2. Mô hình mạng nơron

Qua quá trình thử nghiệm mô hình mạng nơron với nhiều giá trị, nhóm tác giả đưa ra kết quả mô hình với 14 node lớp input, 128 node trong lớp ẩn 1, 64 node trong lớp ẩn 2 và cuối cùng là nhãn output có 5 nhãn. Với các giá trị này, quá trình thực nghiệm đạt kết quả tốt nhất. Ở đây, để tăng hiệu quả mô hình và giảm overfiting trong quá trình huấn luyện, ta dùng kỹ thuật Dropout.

Sau khi khởi tạo mạng nơron, chúng ta tiến hành huấn luyện mô hình:

```
print("Bắt đầu huấn luyện mô hình Ne
ural Network")

start = time()

history = model.fit(X_train, y_train
_encode,validation_split = 0.2, shuf
fle = True, batch_size=256, epochs=1
0, callbacks=[stopping])

end = time()
```

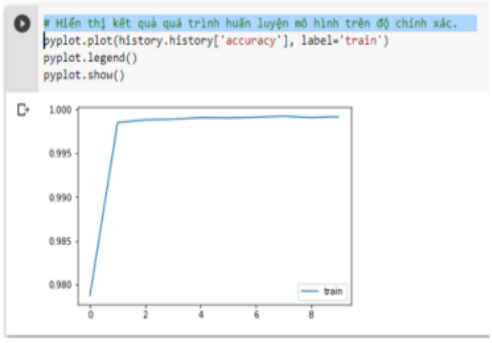
```
print("Thời gian huấn luyện mô hình là: ", end - start)
Bắt đầu huấn luyện mô hình Neural Network
Train on 2347853 samples, validate on 586964 samples
Epoch 1/10
2347853/2347853 [==============================] - 51s 22us/step - loss: 0.5065 - accuracy: 0.9777 - val_loss: 0.1847 - val_accuracy: 0.9992
```

Mô hình được huấn luyện đến epoch thứ 10 thì đạt được sự ổn định và độ chính xác mô hình là tốt nhất:

```
Epoch 7/10
2347853/2347853 [==============================] - 50s 21us/step - loss: 0.3833 - accuracy: 0.9990 - val_loss: 0.2230 - val_accuracy: 0.9998
Epoch 8/10
2347853/2347853 [==============================] - 49s 21us/step - loss: 0.4331 - accuracy: 0.9989 - val_loss: 0.2013 - val_accuracy: 0.9999
Epoch 9/10
2347853/2347853 [==============================] - 49s 21us/step - loss: 0.6322 - accuracy: 0.9988 - val_loss: 0.5474 - val_accuracy: 0.9999
Epoch 10/10
2347853/2347853 [==============================] - 50s 21us/step - loss: 0.8242 - accuracy: 0.9989 - val_loss: 0.4445 - val_accuracy: 0.9999
Thời gian huấn luyện mô hình là: 499.8155708312988
```

Trong đó, epoch là số lần lặp lại và batch_size là số lượng mẫu, hai giá trị này được tính theo công thức đã cho. Sau mỗi lặp sẽ tiến hành drop out, chỉnh sửa mô hình để có thể đạt được loss thấp nhất, accuracy cao nhất và sử dụng đồ thị để hiển thị kết quả huấn luyện dựa trên độ chính xác.

Kết quả huấn luyện đạt được độ chính xác trung bình là 99.99% với thời gian huấn luyện trung bình là 49.98 s. Đây là kết quả tốt cho mô hình phát hiện mạng Botnet. Dưới đây là đồ thị mô tả kết quả huấn luyện.

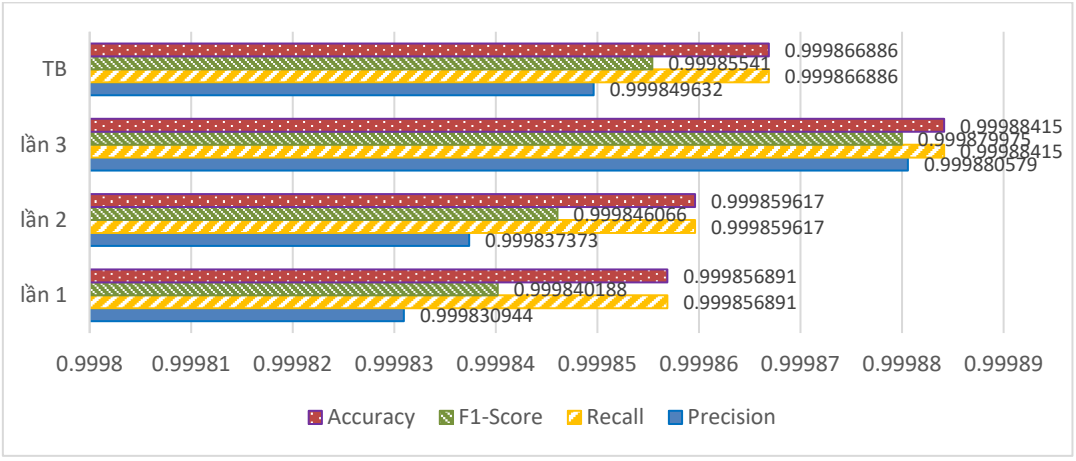


Hình 3. Kết quả huấn luyện

3. KẾT QUẢ VÀ THẢO LUẬN

3.1. Đánh giá mô hình

Để việc đánh giá được đơn giản và hiệu quả, tác giả đã tiến hành đánh giá mô hình dựa theo bốn độ đo phổ biến đó là: Precision, Recall, F1-Score và Accuracy. Các độ đo này có điểm bằng giá trị trung bình của từng nhãn. Việc đánh giá sẽ được thực hiện bằng cách cho chạy thuật toán ba lần và tính ra kết quả (Hình 4)



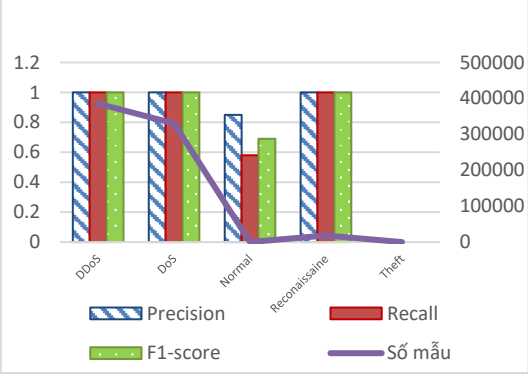
Hình 4. Mô hình hóa đánh giá dạng cột

Như vậy, ta có thể thấy được mô hình đã xây dựng dự đoán chính xác với tỉ lệ ở bốn độ đo rất đồng đều nhau. Qua đó, ta rút ra được kết luận: Với Precision là 0.9997 cho thấy cứ 10000 trường hợp dương tính (là Botnet) dự đoán được thì có 9997 trường hợp là dương tính thật; Recall là 0.9996 cho thấy cứ 10000 trường hợp dương tính thật sự thì mô hình dự đoán ra được 9996 trường hợp dương tính trong số đó. Còn lại hai độ đo F1 và Accuracy đều rất cao (0.9998 và 0.9996) cũng đã cho thấy hiệu quả của mô hình trong phân biệt Botnet và bình thường.

Để việc đánh giá trực quan hơn nữa, tác giả đã tính giá trị độ đo đối với từng nhãn: Precision, Recall và F1-score và đưa ra kết quả đối với từng phần như sau:

Bảng 2. Kết quả đối với từng phần

	Precision	Recall	F1-score	Số mẫu	Precision
DdoS	1.00	1.00	1.00	385309	
DoS	1.00	1.00	1.00	330112	
Normal	0.85	0.69	0.69	107	
Reconaissaine	1.00	1.00	1.00	18163	
Theft	0.00	0.00	0.00	14	

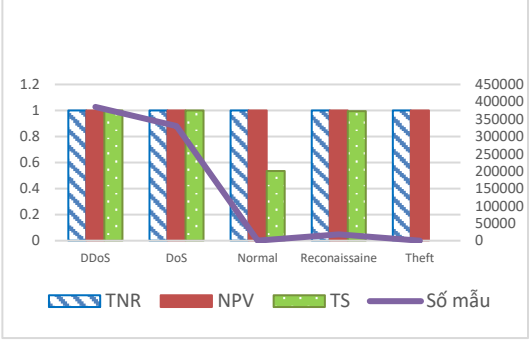


Hình 5. Mô hình hóa kết quả từng phần

Các tham số của Bảng 3 được xác định như sau: TNR (Selectivity – Tỉ lệ giá trị âm tính dự đoán chính xác trên giá trị âm tính thực sự), NPV (Tỉ lệ giá trị âm tính dự đoán chính xác trên giá trị âm tính dự đoán được) và TS (Threat Score – cũng giống như F1-score sẽ là giá trị tổng hợp giữa Precision (PPV) và Recall (TPR)).

Bảng 3. Kết quả các độ đo còn lại

	TNR	NPV	TS	Số mẫu
DdoS	0.999966	0.999871	0.999852	385309
DoS	0.999975	0.99997	0.999933	385309
Normal	0.999937	0.99999	0.535088	107
Reconai-ssaine	0.999943	0.999918	0.994512	18163
Theft	0.999981	1	0	14



Hình 6. Mô hình hóa các độ đo còn lại

Từ các độ đo đối với từng thành phần ta rút ra được nhận xét là khi số lượng mẫu dương tính của loại đó lớn thì việc dự đoán độ chính xác các giá trị âm tính, dương tính đối với loại đó sẽ chính xác hơn. Còn khi số mẫu dương tính nhỏ, ví dụ như loại Normal thì viêc dự đoán sẽ không còn chính xác. Tuy nhiên, do sự chênh lệch giữa số lượng mẫu dương tính và âm tính là quá lớn nên các độ đo là TNR và NPV vẫn cho kết quả cao, tương tự đối với trường hợp của Theft. Do đó, việc đánh giá khả năng hoạt động của mô hình sẽ chính xác nhất khi dựa vào các độ đo như: Precision, Recall và F1-score.

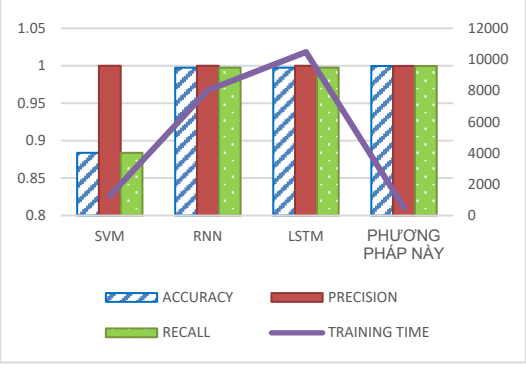
3.2. So sánh các phương pháp

Sau khi xây dựng và chạy thực nghiệm mô hình mạng nơron thành công với các giá trị tham số đã đề xuất, để tiến hành làm rõ kết quả của phương pháp này, nhóm tác giả đã tiến hành chạy thực nghiệm ba phương pháp học máy khác bao gồm SVM (Support Vector Machine), RNN (Recurrent Neural Network – mạng nơron hồi quy) và LSTM (Long Short Term Memory) đã được giới thiệu (Koroniotis và cs., 2019) với phương pháp học sâu.

Sau khi cài đặt và chạy thực nghiệm thành công cả bốn phương pháp: SVM, RNN, LSTM, DL (mô hình mạng nơron tự đề xuất) thì nhóm tác giả tiến hành so sánh, đánh giá mô hình, kết quả như sau:

Bảng 4. So sánh các phương pháp

	SVM	RNN	LSTM	DL
Accuracy	0.88372702	0.99740468	0.99741 94	0.999611657
Precision	1	0.99990435	0.99991036	0.999611657
Recall	0.8837119	0.99749976	0.99991036	0.999631538
Traning Time	1270.48	8035	10482.1 9	499.8155 7083



Hình 7. So sánh với các phương pháp khác

Bảng 4 cho thấy kết quả của mô hình mạng nơron đề xuất (DL) tốt hơn hẳn so với các phương pháp khác: Độ chính xác là 99.9% tốt hơn so với SVM là 88.3% và gần tương đương với RNN, LSTM là 99.7%. Đồng thời, thời gian huấn luyện của mô hình ngắn hơn 1/3 so với SVM, 2/3 so với RNN và 1/2 so với LSTM.

Như vậy, khi so sánh với các phương pháp khác, phương pháp học sâu đã được áp dụng đem lại kết quả cao hơn, thời gian huấn luyện ngắn hơn. Cũng từ Bảng 4 ta có thể thấy được, hai phương pháp có sử dụng mạng nơron khác là RNN và LSTM cũng đem lại kết quả cao so với phương pháp SVM. Tuy nhiên, Precision của SVM là 1 cho thấy rằng phương pháp SVM dự đoán rất chính xác các trường hợp dương tính, không dự đoán nhầm âm tính thành dương tính, nhưng nó lại có nhược điểm là dễ để sót khá nhiều trường hợp dương tính.

Vì vậy, có thể rút ra kết luận là phương pháp SVM sẽ tốt hơn các phương pháp học sâu trong việc dự đoán các bộ dữ liệu nhỏ hoặc các bộ dữ liệu ít trường hợp dương tính. Tuy nhiên, trong phạm vi dữ liệu điều tra mạng (Network Forensics) thì rõ ràng tính tối ưu của các phương pháp học sâu sẽ được thể hiện rõ ràng hơn.

Với mô hình đã xây dựng, chúng ta có thể sử dụng các chức năng cơ bản đó là: Import tập các record, đầu ra sẽ cho biết có bao nhiêu lưu lượng Botnet trong đó, các loại tấn công đã được thực hiện, các địa chỉ Ip nguồn và xuất một file csv chỉ ra các luồng mạng là Botnet. Tuy nhiên, để áp dụng trong mô hình mạng thực tiễn đang được triển khai là không khả thi vì bộ dữ liệu phải được cập nhật Real-time, tốc độ xử lý phải nhanh và có thể đưa ra kết quả trả về ngay khi phát hiện.

4. KẾT LUẬN

Bài báo đã rút ra được vai trò của học sâu trong phát hiện mạng Botnet và thực hiện phát hiện mạng Botnet dựa trên học sâu. Bài báo đã nêu ra hai phương pháp phát hiện Botnet dựa trên học sâu đó là BoTShark-SA và BoTShark-CNN dựa trên hai kỹ thuật Autoencoder và CNN.

Trong bài báo, nhóm tác giả đã đề xuất mô hình mạng nơron với 14 node lớp input, 128 node trong lớp ẩn 1, 64 node trong lớp ẩn 2 và cuối cùng là nhãn output có 5 nhãn và đã so sánh kết quả với ba mô hình học máy khác là SVM, RNN, LSTM. Kết quả nghiên cứu cho thấy tính hiệu quả và ưu việt của mô hình học sâu phát hiện Botnet.

Trong thời gian tới, nhóm tác giả mong muốn trực tiếp thực thi trên tập dữ liệu thô sau khi chặn bắt được dữ liệu mạng, qua đó giảm bớt thời gian xử lý dữ liệu và tăng cường khả năng ứng dụng của mô hình để có thể áp dụng vào thực tiễn theo thời gian thực (Real-time). Xây dựng ứng dụng học sâu phát hiện Botnet có thể đóng góp vào nhiệm vụ bảo đảm an ninh an toàn thông tin, bảo vệ bí mật Nhà nước trong lực lượng Công an nhân dân.

TÀI LIỆU THAM KHẢO

Abu Rajab, M., Zarfoss, J., Monrose, F., & Terzis, A. (2006). A multifaceted approach to understanding the botnet phenomenon. *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, 41–52. https://doi.org/10.-1145/1177080.1177086

Đoàn Xuân Dũng. (2018). *Tóm tắt văn bản sử dụng các kỹ thuật trong Deep Learning* [Luận văn thạc sỹ, Trường Đại học Công nghệ, Đại học Quốc gia Hà Nội]. http://repository.vnu.edu.vn/handle/VNU_123/65708

Guntuku, S. C., Narang, P., & Hota, C. (2013). *Real-time Peer-to-Peer Botnet Detection Framework based on Bayesian Regularized Neural Network* (arXiv:1307.7464). arXiv. https://doi.-org/10.48550/arXiv.1307.7464

Haddadi, F., & Zincir-Heywood, A. N. (2017). Botnet behaviour analysis: How would a data analytics-based system with minimum a priori information perform? *International Journal of Network Management*, *27*(4), e1977. https://doi.-org/10.1002/nem.1977

Kirubavathi, G., & Anitha, R. (2016). Botnet detection via mining of traffic flow characteristics. *Computers & Electrical Engineering*, *50*, 91–101. https://doi.org-/10.1016/j.compeleceng.2016.01.012

Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. *Future Generation Computer Systems*, *100*, 779–796. https://doi.org/10.-1016/j.future.2019.05.041

Stevanovic, M., & Pedersen, J. M. (2014). An efficient flow-based botnet detection using supervised machine learning. *2014 International Conference on Computing, Networking and Communications (ICNC)*, 797–801. https://doi.org/10.-1109/ICCNC.2014.6785439

Trần Thị Hằng. (2020). *Nghiên cứu tìm hiểu thực trạng về an ninh mạng và biện pháp khắc phục* [Luận văn thạc sỹ, Trường Đại học Dân lập Hải Phòng].

Wang, J., & Paschalidis, I. Ch. (2017). Botnet Detection Based on Anomaly and Community Detection. *IEEE Transactions on Control of Network Systems*, *4*(2), 392–404. https://doi.org-/10.1109/TCNS.2016.2532804