SỬ DỤNG KIỂM ĐỊNH GIẢ THUYẾT BAYES VÀ NEYMAN-PEARSON CHO BỘ TỰ MÃ HÓA ĐỂ PHÁT HIỆN BẤT THƯỜNG TRONG AN NINH MẠNG

NGUYỄN VĂN ANH TUẤN*, ĐINH HOÀNG HẢI ĐĂNG, TRẦN NAM BÁ,
NGUYỄN THỊ THANH HÒA, TRỊNH THỊ BẢO BẢO, PHAN LÊ HOÀNG VIỆT,
NGUYỄN CHÍ KIÊN, NGUYỄN HỮU TÌNH

Khoa Công nghệ Thông tin, Đại học Công nghiệp Thành phố Hồ Chí Minh

* *Tác giả liên hệ: nvatuan3@gmail.com*

DOIs: https://doi.org/*10.46242/jstiuh.v61i07.4724*

Tóm tắt. Bộ tự mã hóa là một mô hình học không giám sát trong đó các tham số được điều chỉnh để vector đầu ra gần giống nhất với vector đầu vào. Trong bài báo này, chúng tôi sử dụng bộ tự mã hóa để phát hiện các kết nối bất thường trong mạng Internet. Mức lỗi tái tạo khi sử dụng bộ tự mã hoá sẽ được sử dụng để phân lớp kết nối thành kết nối bình thường và kết nối bất thường. Chúng tôi trình bày ba phương pháp phân lớp độ lỗi tái tạo: phân lớp sử dụng một ngưỡng cho trước, phân lớp theo kiểm định giả thuyết Bayes và phân lớp theo kiểm định giả thuyết Neyman-Pearson. Độ chính xác trung bình đạt được trên ba phương pháp là $96.65 \pm 0.98\%$ trên bộ dữ liệu NSL KDD.

Từ khóa: Bộ tự mã hóa, kiểm định giả thuyết Bayes, kiểm định giả thuyết Neyman-Pearson, phát hiện bất thường.

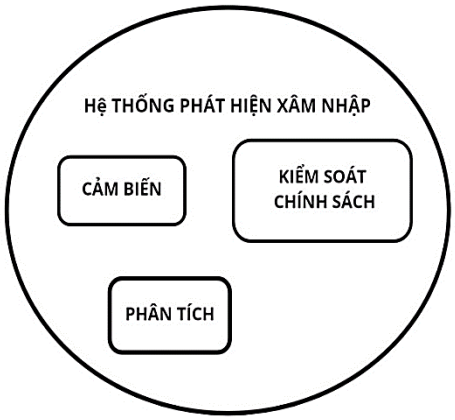
1. GIỚI THIỆU

Trong thời đại kỹ thuật số phát triển như hiện nay, việc truy cập mạng bằng Internet trở nên rất dễ dàng đối với bất kì người dùng cuối nào; chỉ cần thiết bị được kết nối với mạng Internet thì việc trao đổi các gói tin với nhau có thể bắt đầu. Với việc sử dụng Internet ngày càng dễ dàng trong cuộc sống ngày nay, lĩnh vực an ninh mạng đã trở thành nền tảng quan trọng cho tất cả các ứng dụng web như đấu giá trực tuyến, bán lẻ trực tuyến... Kéo theo đó, sự phát triển của phần mềm độc hại với mục đích tấn công mạng đặt ra một thách thức lớn đối với việc thiết kế các hệ thống phát hiện xâm nhập. Các cuộc tấn công mạng ngày càng trở nên tinh vi hơn và tạo ra thách thức hàng đầu là xác định chính gói tin đó có độc hại hay là không.

Phát hiện xâm nhập là việc cố gắng phát hiện các cuộc tấn công của máy tính bằng cách kiểm tra các dữ liệu khác nhau hay sự bất thường giữa các gói tin được giám sát trong các quá trình truyền tải gói tin giữa các mạng. Đây có thể coi là một trong những hướng tiếp cận quan trọng để giải quyết các vấn đề an ninh mạng một cách hiệu quả. Một hệ thống phát hiện xâm nhập thông thường cần có ba thành phần chính: Cảm biến/đầu dò, bàn điều khiển phân tích, kiểm soát/phản hồi chính sách [1]. Hình 1 minh họa hệ thống này.

Có hai phương pháp chính để phát hiện xâm: Các phương pháp sử dụng quy tắc (rule-based methods) và các phương pháp phát hiện bất thường (anomaly detection methods). Trong bài báo này chúng tôi sẽ tập trung vào phương pháp tiếp cận phát hiện bất thường. Cụ thể, chúng tôi tập trung vào phương pháp tiếp cận chính là mô hình học không giám sát với bộ tự mã hóa kết hợp với các phương pháp phân lớp để phát hiện bất thường.

2. CÁC NGHIÊN CỨU LIÊN QUAN



Hình 1: Khuôn khổ tổng quát của hệ thống phát hiện xâm nhập.

Các hệ thống phát hiện xâm nhập đã và đang là một bài toán đang được nghiên cứu và phát triển để tạo ra một hệ thống phát hiện hiệu bảo vệ hệ thống mạng. Dưới đây chúng tôi khảo sát một số nghiên cứu trước đây trong lĩnh vực này.

Bài nghiên cứu của [2] về một hệ thống tích hợp hai kỹ thuật học sâu và học nông, sử dụng 2 thuật toán bộ tự mã hóa thưa để trích xuất đặc trưng và giảm chiều dữ liệu, kết hợp vector học máy hỗ trợ để phân lớp. Đồng thời bài nghiên cứu cũng đề cập đến sự cải thiện của mô hình so với các mô hình học nông cơ bản (SVM, Naïve Bayes, Random Forest,...) trên tập dữ liệu NSL-KDD. Kết quả cho thấy hệ thống kết hợp mà các nhà nghiên cứu đề xuất có sự cải thiện rõ rệt về thời gian huấn luyện cũng như thời gian thực hiện, độ chính xác được cải thiện nhưng không đáng kể (~5%).

Trong khi đó các tác giả của [3] so sánh độ hiệu quả của các loại bộ tự mã hóa khác nhau trong bài toán phân lớp đơn lớp. Trong bài báo, tác giả đã xây dựng 5 loại bộ tự mã hóa khác nhau để so sánh, lần lượt là bộ tự mã hóa xếp chồng, bộ tự mã hóa thưa thớt, bộ tự mã hóa khử nhiễu, bộ tự mã hóa tương phản (contrastive) và bộ tự mã hóa tích chập. Chúng tôi sử dụng kết quả thu được từ bài báo này làm giá trị tham khảo để kiểm tra độ chính xác và tính đúng đắn của mô hình phát hiện bất thường của chúng tôi.

Trong bài báo [4], tác giả đề xuất sử dụng phân tích thống kê và bộ tự mã hóa để phân lớp bao gồm phân lớp hai lớp và phân lớp nhiều lớp. Và so sánh hiệu suất với với các mô hình phân lớp khác. Bộ dữ liệu NSL KDD được sử dụng để đánh giá hệ thống phát hiện xâm nhập. Kết quả cho thấy bộ tự mã hóa được đề xuất có hiệu suất tốt hơn tất cả các cách tiếp cận khác.

Trong bài báo [5], tác giả đã đề xuất mô hình phát hiện xâm nhập mạng dựa trên các dilated convolutional autoencoder xếp chồng. Mô hình thực hiện trên 2 bộ dữ liệu là bộ dữ liệu CTU-UNB và bộ dữ liệu Contagio-CTU-UNB. Kết quả thử nghiệm cho thấy mô hình phát hiện hiệu quả các cuộc tấn công phức tạp từ rất nhiều dữ liệu không có nhãn. Hiệu xuất này có thể đáp ứng các yêu cầu của môi trường mạng quy mô lớn và thức tế. Bên cạnh đó, mô hình cũng có hạn chế là quá trình đào tạo mất một lượng thời gian lớn.

Trong bài báo [6], nhóm tác đã đánh giá kết hợp các cấu trúc mô hình bộ tự mã hóa khác nhau được kiểm thử trên các bộ dữ liệu chuẩn NSL-KDD, IoTID20 và BaIoT để xác định kiến trúc mô hình tối ưu nhằm cung cấp hiệu quả tốt nhất. Trong nghiên cứu, mô hình đã cho điểm F1 tốt nhất là 0.894 với bộ dữ liệu NSL-KDD. Kết quả thử nghiệm cho thấy rằng kích thước mô hình và kích thước của lớp nút cổ chai (bottle neck) có ảnh hưởng đến hiệu suất của IDS.

Ở bài báo [7],tác giả đã sử dụng bộ tự mã hóa để giải quyết 2 nhiệm vụ an ninh mạng khác nhau: phát hiện xâm nhập bất thường và phân lớp phần mềm độc hại. Cụ thể, bài báo này chỉ ra rằng việc sử dụng 2 kĩ thuật để huấn luyện đó là stacked Restricted Boltz-mann Machine (RBM) và bộ tự mã hóa khử nhiễu xếp chồng. Đối với phân lớp phần mềm độc hại, tác giả sử dụng bộ dữ liệu Microsoft Malware Classification Challenge (BIG2015) được lưu trữ tại Kaggle, đối với phát hiện xâm nhập bất thường tác giả sử dụng NSL-KDD. Kết quả cho thấy các mô hình KNN, Gaussian Naive Bayes và SVM để phân lớp khi sử dụng tập thuộc tính được tái tạo bằng bộ tự mã hóa có cải thiện về độ chính xác mô hình đáng kể. Tuy nhiên đối với phân lớp

phần mềm độc hại, Xgboost có thể phân lớp trên tập dữ liệu Unigram ban đầu tốt hơn so với khi sử dụng tập dữ liệu tái tạo bằng bộ tự mã hóa.
Trong bài báo [8], tác giả đề cập đến cập đến vấn đề phát hiện bất thường không dễ thực hiện với thời gian và không gian phức tạp lớn. Từ đó, một phương pháp kiểm định giả thuyết đã được đề xuất bằng cách thu thập một bộ mẫu quan sát. Thứ nhất nó có thể tạo ra các luật bằng các quy tắc suy luận Bayes. Sau đó, nó có thể phát hiện tình hình an ninh mạng có bình thường hay không bằng cách kiểm tra giả thuyết, từ đó có thể dự báo xu hướng của tình hình an ninh mạng trong tương lai. Nhờ đó, nó có thể cung cấp cơ sở đáng tin cậy để nhà quản trị đưa ra các quyết định và các biện pháp phòng vệ. Cuối cùng, việc sử dụng toàn bộ dữ liệu mô phỏng, thuật toán phát hiện bất thường mạng về tình hình an ninh được xác minh và kết quả cho thấy phương pháp này là đúng và khả thi.
Ở bài toán phát hiện bất thường trong âm thanh của bài báo [9], tác giả trình bày một phương pháp khá tương tự chúng tôi trong việc huấn luyện và phát hiện bất thường. Bằng cách sử dụng mô hình bộ tự mã hóa học không giám sát và huấn luyện trên các âm thanh bình thường, và đem đi phát hiện các âm thanh bất thường. Tuy nhiên, việc huấn luyện không bao gồm các âm thanh bất thường sẽ không thực sự giảm tỉ lệ dương tính đúng (TPR), tác giả đã sử dụng bổ đề Neyman-Pearson để biến bài toán thành một bài toán kiểm định giả thuyết, tạo một hàm mục tiêu mới để đi tối đa hóa tỉ lệ dương tính đúng khi vẫn trong điều kiện tỉ lệ âm tính sai (FPR) thấp. Phương pháp được thử nghiệm trên các âm thanh được tổng hợp, cải thiện tỉ lệ phát hiện trong điều kiện FPR thấp, và cam đoan có thể sử dụng được để phát hiện âm thanh bất thường trong điều kiện thực tế.

3. PHƯƠNG PHÁP TIẾP CẬN

3.1. Bộ tự mã hóa

3.1.1. Tổng quan về bộ tự mã hóa
Bộ tự mã hóa là một dạng mạng thần kinh nhân tạo được dùng để học cách mã hóa dữ liệu hiệu quả sử dụng cách học không giám sát. Bộ tự mã hóa học các dữ liệu đầu vào và cố gắng tạo ra dữ liệu đầu ra gần giống với dữ liệu đầu vào nhất. Một bộ tự mã hóa bao gồm 2 phần đó là bộ mã hóa (encoder) và bộ giải mã (decoder) [10]. Bộ mã hóa nén dữ liệu đầu vào vào một chiều thấp hơn, còn bộ giải mã xây dựng lại dữ liệu có số chiều bằng với số chiều của dữ liệu đầu vào dựa vào chiều thấp hơn đó.
Đối với tập dữ liệu X đào tạo gồm m cột $X = x_1, x_2, \ldots, x_m$, với mỗi x_i là 1 vector d chiều, bộ mã hóa sẽ ánh xạ mỗi vector x_i vào 1 vector h_i qua phép ánh xạ f_θ như dưới đây:

$$h_i = f_\theta(x_i) = s(Wx_i + b) \quad (1)$$

trong đó, W là ma trận $d \times d$ chiều, với d là số nơ ron trong lớp ẩn (b là vector hệ số chặn), θ là tập tham số ánh xạ $\theta = \{W, b\}$, s là hàm kích hoạt được định nghĩa là hàm sigmoid như sau:

$$s(t) = \frac{1}{1 + e^{-t}} \quad (2)$$

Trong đó tham số t ảnh hưởng đến hình dạng hàm. Bộ giải mã ánh xạ mỗi h_i từ hidden layer thành 1 vector x_i' có d chiều qua phép ánh xạ $g_{\theta'}$:

$$x_i' = g_{\theta'}(x_i) = s(W'h_i + b') \quad (3)$$

trong đó, W' là ma trận $d \times d$ chiều, với d là số nút trong lớp ẩn, b' là vector hệ số chặn (bias vector), θ' là tập tham số ánh xạ $\theta' = \{W', b'\}$ [11].
Mục tiêu đào tạo bộ tự mã hóa là mô phỏng được dữ đầu vào 1 cách chi tiết nhất, sao cho độ sai số (lỗi tái tạo) giữa đầu vào (x) và đầu ra (x') là nhỏ nhất, hay có thể biểu diễn hàm mất mát dưới dạng công thức dưới đây.

$$L(x, x') = \frac{1}{m}\sum_{i=1}^{m} ||x_i - x_i'||^2 \quad (4)$$

Để tối ưu hóa bài toán, ta cần là tìm ra $\theta^\star$ và $\theta'^\star$, là bộ trọng số tối ưu nhất, hay:

$$\theta^\star, \theta'^\star = \underset{\theta,\theta'}{\operatorname{argmin}} L(x, x') \quad (5)$$

3.1.2. Dùng bộ tự mã hóa khử nhiễu (Denoising Autoencoder) trong việc phát hiện xâm nhập

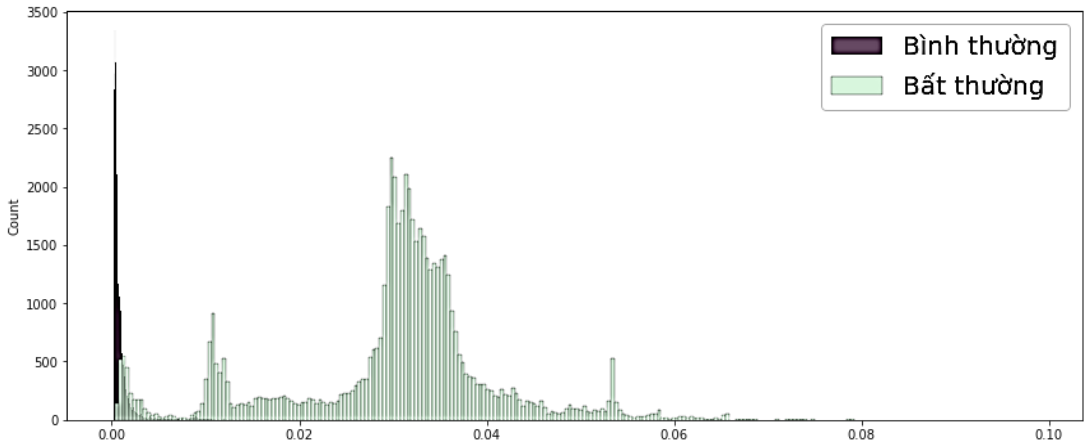
Hệ thống của chúng tôi sử dụng bộ tự mã hóa khử nhiễu gồm 2 giai đoạn: Giai đoạn huấn luyện và giai đoạn kiểm thử. Trong giai đoạn huấn luyện, dữ liệu huấn luyện chỉ bao gồm dữ liệu có nhãn bình thường, giai đoạn kiểm thử chúng tôi sử dụng dữ liệu có nhãn bao gồm cả bình thường và bất thường, vì đầu ra của bộ tự mã hóa khử nhiễu mô tả lại gần đúng nhất bộ dữ liệu tập huấn luyện, nên khi đó, sai số giữa dữ liệu đầu vào và dữ liệu đầu ra của các dữ liệu mang nhãn bất thường sẽ cao hơn sai số của các dữ liệu mang nhãn bình thường. Để tính toán sai số này, chúng tôi sử dụng sai số tuyệt đối trung bình, sau đó sử dụng tập sai số này để phân lớp dữ liệu.

Về cấu trúc cụ thể của bộ tự mã hóa, kích cỡ của lớp mã hóa (encoder) và lớp giải mã (decoder) có một lớp cao hơn kích cỡ của lớp đầu vào (input) và đầu ra (output). Lớp mã hóa với 128 - 64 - 32 và lớp giải mã là 32 - 64 - 128. Lớp nút cổ chai (bottle neck) có kích cỡ nhỏ nhất là 16, giúp lưu trữ những thông tin quan trọng nhất của tập dữ liệu. Hai lớp đầu vào và đầu ra có kích cỡ giống nhau là 146. Hình 2 thể hiện cấu trúc của bộ tự mã hóa được sử dụng trong bài báo.

Sau khi tái cấu trúc, chúng tôi nhận thấy rằng phân phối lỗi của nhãn bình thường và bất thường khác biệt rõ rệt (mặc dù vẫn có một lượng dữ liệu bị trùng phân phối lỗi), hình 3 mô tả phân phối lỗi này.

Vì vậy chúng tôi sử dụng lỗi tái cấu trúc này để phân lớp bằng cách áp dụng 3 phương pháp phân lớp, đó là: dùng một ngưỡng để phân lớp, dùng kiểm định giả thuyết Bayes và dùng kiểm định giả thuyết Neyman-Pearson.

3.2. Các phương pháp để phân lớp

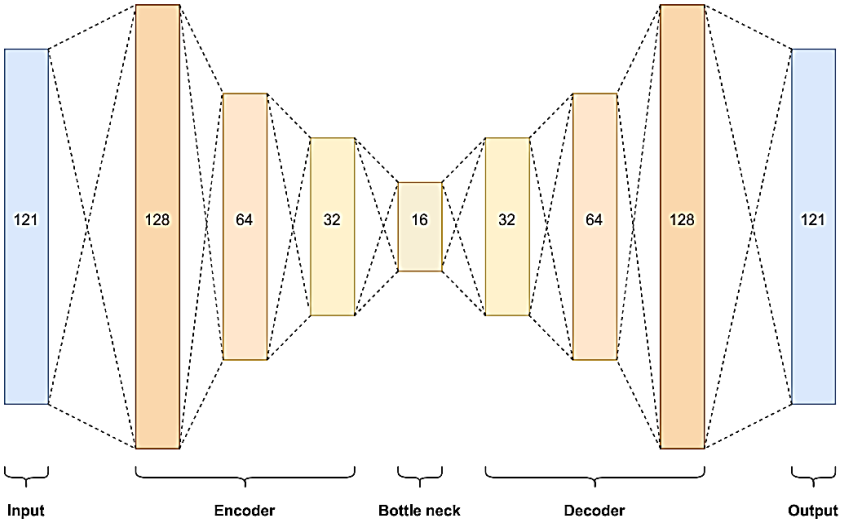


Hình 3: Biểu đồ tần suất lỗi tái tạo

3.2.1. Dùng một ngưỡng để phân lớp

Trong bài báo [12], tác giả đã trình bày phương pháp phân lớp bình thường và bất thường từ tập lỗi tái tạo. Cụ thể là sau khi xây dựng mô hình, tác giả đã tính ngưỡng θ_α theo công thức (5).

$$\theta_\alpha = \mu_\alpha + Z_\alpha \sigma_\alpha \quad (6)$$



Hình 2: Sơ đồ các lớp lớp của bộ tự mã hóa khử nhiễu được sử dụng trong bài báo

cho các tập dữ liệu của tác giả (tác giả chia 3 tập dữ liệu với tỉ lệ dữ liệu bình thường và bất thường khác nhau nhằm mục đích đánh giá), trong đó giá trị μ_α là giá trị trung bình của toàn bộ tập lỗi tái tạo, Z_α là trị thống kê Z cho ngưỡng trên $\alpha\%$ của phân phối chuẩn tắc, và σ_α là độ lệch chuẩn của toàn bộ tập lỗi tái tạo. Trong đó, toàn bộ tập lỗi tái tạo có nghĩa là bao gồm cả tập dữ liệu bình thường được huấn luyện và tập bất thường.

Trong phạm vi bài toán này và qua quá trình thử nghiệm, chúng tôi cố định $\alpha = 0.38$. Kết quả thực nghiệm được đưa ra trong phần 4.4.

3.2.2. Kiểm định giả thuyết Bayes

Kiểm định giả thuyết ngày càng được sử dụng rộng rãi trong an ninh mạng để phát hiện các cuộc tấn công bất thường và các hành vi độc hại. Tổng quan về các phương pháp tiếp cận khác nhau để kiểm tra nhiều giả thuyết được cung cấp, trong số đó có kiểm định giả thuyết Bayes, kiểm định giả thuyết minimax, centralized,... [13] Với kiểm định giả thuyết Bayes, người ta giả định rằng xác suất tiên nghiệm của M giả thuyết là $\pi_0, \pi_1, ..., \pi_{M-1}$. Và ma trận chi phí $\{C_{ij}\}$ được định nghĩa để đánh giá độ rủi ro của giả thuyết trong bài toán [14]. Ở nghiên cứu này, chúng tôi chỉ đề cập hai giả thuyết H_0 và H_1.

Gọi H_0 là phân phối lỗi tái tạo của lớp bất thường, H_1 là phân phối lỗi tái tạo của lớp bình thường, và Y là các điểm dữ liệu. Giả sử biết trước các xác suất $P(H_0) = p_0, P(H_1) = p_1$ và $p_0 + p_1 = 1$. Lúc này, phân phối Y của 2 giả thuyết trên là:

$$f_Y(y|H_0), f_Y(y|H_1) \tag{7}$$

Áp dụng Bayes, ta có:

$$P(H_0|Y = y) = \frac{f_Y(y|H_0)P(H_0)}{f_Y(y)} \tag{8}$$

$$P(H_1|Y = y) = \frac{f_Y(y|H_1)P(H_1)}{f_Y(y)} \tag{9}$$

Ta có thể định nghĩa ma trận $\{C_{ij}\}$ như sau: $C_{00} = 0$ là rủi ro khi dự đoán bất thường là bất thường, $C_{11} = 1$ là rủi ro khi dự đoán bình thường là bình thường, $C_{01} = 1$ là rủi ro khi dự đoán bất thường là bình thường và $C_{10} = 0.1$ là rủi ro khi dự đoán bình thường là bất thường. Mục đích của kiểm định giả thuyết bayes là tối thiểu hóa trung bình kì vọng $\mathrm{E}[C_{ij}]$.

$$r(\delta) = \sum_{j=0}^{M-1} \pi_j R_j(\delta) \tag{10}$$

Trong đó, $R_j(\delta)$ là rủi ro có điều kiện đối với quy tắc quyết định δ khi biết trước H_j, hay

$$R_j(\delta) = \sum_{j=0}^{M-1} C_{ij} P_j(\delta(y) = i) \tag{11}$$

Trong đó $P_j(\delta(y=i))$ là xác suất của một quy tắc quyết định cụ thể δ được dự đoán là H_i, trong khi thực tế là H_j. Mặt khác,

$$\delta(y) = \begin{cases} 1 & \text{nếu } \dfrac{P_1(y)}{P_0(y)} \geq \dfrac{\pi_0(C_{10} - C_{00})}{\pi_1(C_{01} - C_{11})} \\ 0 & \text{các trường hợp khác.} \end{cases} \tag{12}$$

Vì bài toán của chúng tôi có $M = 2$ nên lúc bấy giờ việc quyết định xem gói tin có bình thường hay không, ta chỉ cần so sánh tỉ lệ của

$$P(H_0 \mid Y = y) \cdot C_{10} \text{ và } P(H_1 \mid Y = y) \cdot C_{01}. \tag{13}$$

Nếu $P(H_1|Y)C_{01} \geq P(H_0|Y)C_{10}$, ta sẽ coi điểm dữ liệu đó là bình thường, ngược lại sẽ là bất thường.

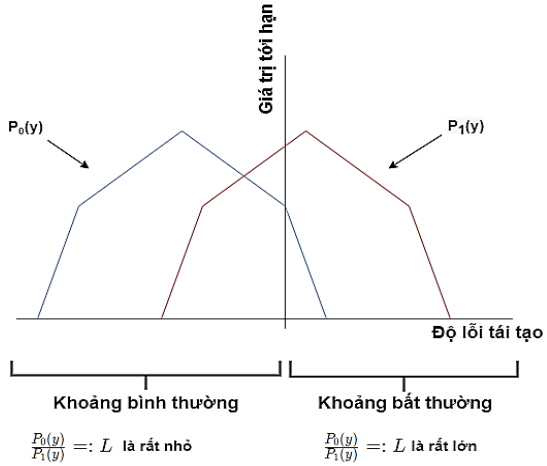
3.2.3. Kiểm định giả thuyết Neyman-Pearson

Bài toán Neyman-Pearson là một trong những bài toán tối ưu hóa xác suất phát hiện (P_D) (hoặc giảm thiểu xác suất bỏ sót $P_M = 1 - P_D$ tuân theo giới hạn trên của xác suất cảnh báo giả (P_F) [15]. Lưu ý rằng:

$$\mathrm{P_F}(\delta) \equiv \mathrm{P_0}(\delta(y) = 1) \tag{14}$$
$$\mathrm{P_D}(\delta) \equiv \mathrm{P_1}(\delta(y) = 1)$$

Vì thế, về mặt toán học, bài toán kiểm định giả thuyết Neyman-Pearson là:

$$\begin{aligned} &\text{Tối thiểu hóa} && \mathrm{P_D}(\delta) \\ &\text{Ràng buộc} && \mathrm{P_F}(\delta) \leq \alpha, 0 < \alpha < 1 \end{aligned} \tag{15}$$



Hình 4: Trực quan vấn đề của bài toán kiểm định giả thuyết Neyman-Pearson

Đặt $\frac{P_0(y)}{P_1(y)} =: L$ là tỉ số khả dĩ, hay còn gọi là tỉ lệ khả năng xảy ra (likelihood ratio), xác suất báo động sai (P_F False Alarm) và xác suất phát hiện (P_D) do quy tắc quyết định (decision rules) định nghĩa có thể được viết là:

$$P_D = P_0(\delta(y) = 0), \tag{16}$$
$$P_F = P_1(\delta(y) = 0) \tag{17}$$

Ta có quy tắc quyết định như sau:

$$\delta(y) = \begin{cases} 0 & \textit{nếu } \dfrac{P_0(y)}{P_1(y)} \geq \gamma \\ 1 & \textit{các trường hợp khác} \end{cases} \tag{18}$$

Giả sử, khi L rất lớn tại đó giá trị xác xuất $P_0(y)$ rất lớn và $P_1(y)$ rất nhỏ. Song, y càng lớn thì đồng nghĩa với việc độ lỗi tái cấu trúc càng lớn nên xác suất là bất thường càng cao, và ngược lại.

Dựa vào phương pháp kiểm định giả thuyết Neyman-Pearson, ý tưởng của chúng tôi sẽ chọn một ngưỡng γ rất lớn, lúc này với quy tắc quyết định gần như luôn trả về $\delta(y) = 1$, tại đó $P_F = P_1(\delta(y) = 0)$ sẽ rất thấp. Sau đó chúng tôi tiến hành giảm dần ngưỡng, lúc này các quyết định $\delta(y) = 0$ sẽ tăng khiến cho $P_F = P_1(\delta(y) = 0)$ tăng dần đến khi $P_F \leq \alpha$ (Với α là ngưỡng tin cậy được chọn ban đầu) và tìm ra ngưỡng γ khi P_D là lớn nhất. Qua thực nghiệm, ngưỡng tin cậy $\alpha = 0.02$ sẽ đưa ra kết quả tốt nhất.

4. ĐÁNH GIÁ

4.1. Bộ dữ liệu NSL_KDD và hướng xử lý

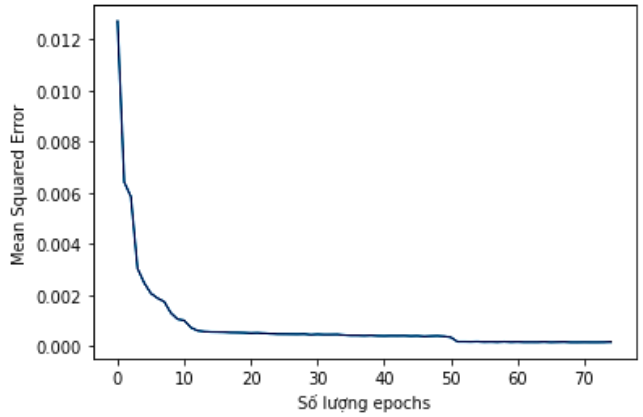
4.1.1. Bộ dữ liệu NSL_KDD

Bộ dữ liệu này được tạo bằng cách thu thập lưu lượng mạng (sử dụng tcpdump) của một hệ thống mạng mô phỏng các loại tấn công khác nhau [16]. Trong khi đó, KDD99 là phiên bản trích xuất đặc trưng của bộ dữ liệu DARPA. NSL-KDD là phiên bản được loại bỏ và giảm kích thước của bộ dữ liệu KDD99. Trong bài báo này chúng tôi sử dụng tập dữ liệu NSL-KDD để huấn luyện và kiểm thử mô hình [17]. Bộ dữ liệu NSL-KDD gồm 2 tập dữ liệu là KDDTrain+ và KDDTest+, tổng cộng gồm có 148516 dòng và 43 cột, tập dữ liệu KDDTrain+ gồm 125973 dòng và 43 cột (tương đương khoảng 85% bộ dữ liệu) và tập dữ liệu KDDTest+ gồm 22543 dòng và 43 cột (tương đương khoảng 15%).

Trong bài báo này, chúng tôi sử dụng tập KDDTrain+ để huấn luyện, dùng KDDTest+ để kiểm thử và lấy kết quả.

4.1.2. Hướng xử lý

Để xử dụng dữ liệu NSL-KDD cho mô hình bộ tự mã hóa, cần đưa dữ liệu về dạng số thay vì như hiện tại là số và chữ (của tập dữ liệu). Hướng xử lý của chúng tôi sẽ bao gồm loại bỏ hai cột *num_outbound_cmds* và *difficult level* vì hai cột này không có ý nghĩa. Sau đó áp dụng mã hóa One-Hot cho các cột là kiểu định tính (kiểu chữ) để dữ liệu cuối cùng thành dạng số, rồi áp dụng chuẩn hóa Min-Max để đưa dữ liệu về khoảng $[0, 1]$, việc chuẩn hóa này áp dụng cho các cột là kiểu số ban đầu và dữ liệu vừa được One-Hot.



Hình 5: Sự thay đổi của độ lỗi trung bình bình phương sai số theo mỗi lần huấn luyện của mô hình

Bên cạnh đó, vì bài toán của chúng tôi là bài toán phân lớp nhị phân, nên dữ liệu bắt buộc chỉ thuộc hai loại. Chúng tôi quy định hai loại này là *bất thường* và *bình thường*. Trong dữ liệu, nhãn *bình thường* sẽ có giá trị là *normal*, còn nhãn *bất thường* sẽ là tất cả nhãn còn lại, bao gồm: *nmap, land, smurf, rootkit, loadmodule, buffer_overflow, pod, back, ipsweep, spy, teardrop, imap, warezmaster, ftp_write, guess_passwd, satan, multihop, portsweep, phf, warezclient, neptune, perl*.

Sau các bước xử lý trên, tập dữ liệu KDDTrain+ trở thành 125973 dòng và 146 cột. Và tập KDDTest+ trở thành 22544 dòng và 146 cột.

4.2. Huấn luyện mô hình Autoencoder

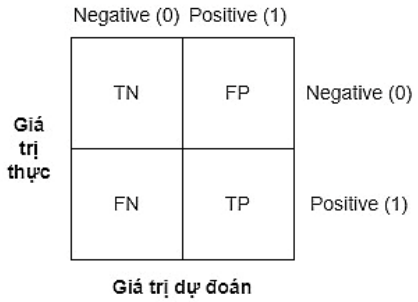
Đối với việc hiện thực mô hình bộ tự mã hóa không giám sát, chúng tôi sử dụng tensorflow làm công nghệ chính để triển khai. Số epochs tối đa chúng tôi để tối đa là 1000, sử dụng mini batchsize với mỗi batch size

là 64, với phương thức đánh giá độ lỗi là trung bình bình phương sai số để tối ưu $L(x, x')$, chúng tôi sử dụng thuật toán tối ưu Adam. Ngoài ra, để tránh việc quá khớp của mô hình, chúng tôi sử dụng dừng sớm. Như trong hình 5 độ lỗi đã giảm xuống dưới mức 0.002 và ổn định trong những epoch 40 trở đi, nên mô hình đã huấn luyện xong trong hơn 70 epochs mà không cần chạy đến tối đa số epochs xác định trước.

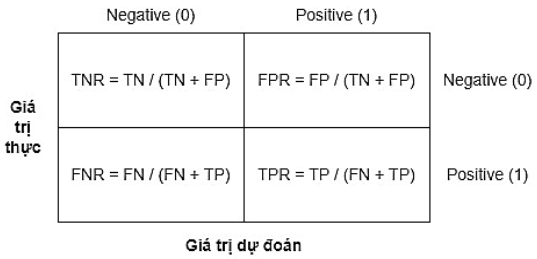
4.3. Các phương pháp đánh giá bộ phân lớp

Bài toán của chúng tôi sử dụng mô hình bộ tự mã hóa không giám sát để tái tạo lại tập dữ liệu được đưa vào, sau đó dùng một trong ba phương pháp được trình bày trong phần 3.2 để phân lớp dữ liệu từ tập lỗi tái tạo đã tính trước. Lúc này, bài toán không giám sát trở thành bài toán phân lớp nhị phân có giám sát với hai lớp là bình thường và bất thường.

Để đánh giá tổng quan kết quả, chúng tôi sử dụng các thang đo: độ chính xác, auc, f1 score và ma trận nhầm lẫn.



Hình 6: Cấu trúc của ma trận nhầm lẫn với 4 giá trị TN, FN, FP và TP.



Hình 7: Cấu trúc của ma trận nhầm lẫn được chuẩn hóa với 4 giá trị TNR, FPR, FNR và TPR

4.3.1. Độ chính xác

Thang đo độ chính xác là thang đo đơn giản nhất, xác định tỉ lệ bao nhiêu giá trị được dự đoán khớp với tập giá trị đã biết. Bộ phân lớp của mô hình càng tốt sẽ cho kết quả accuracy càng cao.

4.3.2. Ma trận nhầm lẫn

Ma trận nhầm lẫn có thể cho chúng ta biết được các thông tin cụ thể hơn về kết quả dự đoán so với thực tế của mô hình [18]. Hình 6 mô tả dạng ma trận nhầm lẫn được sử dụng trong bài toán của chúng tôi. Các ô chứa các số lượng của giá trị tương ứng.

Trong bài toán phát hiện bất thường của chúng tôi, số lượng các giá trị phân lớp chỉ có 2, nên người ta hay dùng Positive (P - dương tính) và Negative (N - âm tính), Negative để kí hiệu cho một lớp có vẻ nghiêm trọng hơn lớp còn lại khi đem vào một bài toán cụ thể nào đó [18], mà trong bài toán này, lớp nghiêm trọng hơn là lớp abnormal - bất thường. Ở hình 6, bốn giá trị được thể hiện lần lượt là True Negative (TN), False Negative (FN), False Positive (FP), và True Positive (TP). Còn ở hình 7, mô tả dạng ma trận nhầm lẫn được chuẩn hóa, kí hiệu R là tỉ lệ (rate).

Phần kết quả sẽ trình bày trên cả hai dạng ma trận nhầm lẫn.

4.3.3. F1 Score

F1 được cấu tạo từ Precision và Recall. Một mô hình phân lớp được đánh giá cao khi cả Precision và Recall đều cao. 2 chỉ số này thấp đều kéo F1 score xuống. Trường hợp xấu nhất khi 1 trong hai chỉ số

Precison và Recall bằng 0 sẽ kéo điểm F1-score về 0. Trường hợp tốt nhất khi cả điểm chỉ số đều đạt giá trị bằng 1, khi đó điểm F-score sẽ là 1. Công thức sẽ là:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{19}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{20}$$

$$\text{F1} = 2\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{21}$$

4.3.4. AUC

Diện tích dưới đường cong (AUC) là thước đo khả năng phân biệt giữa các lớp của một bộ phân lớp và được sử dụng như một bản tóm tắt của đường cong ROC. AUC càng cao, hiệu suất của mô hình càng tốt trong việc phân biệt giữa các lớp *bất thường* và *bình thường*.

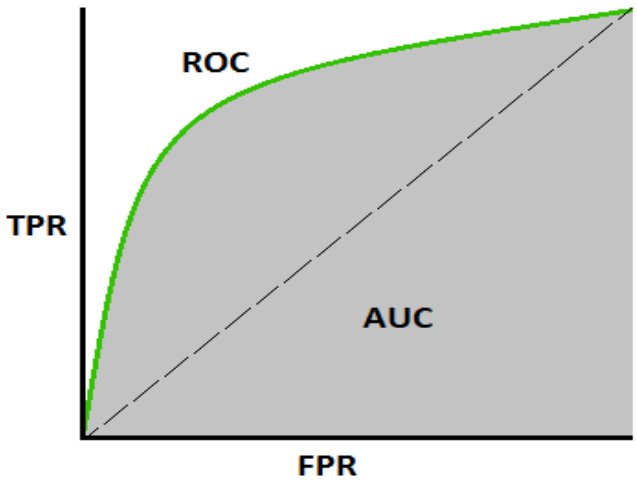
4.4. Kết quả

Để kiểm tra độ tốt của mô hình bộ tự mã hóa học không giám sát, chúng tôi đưa cả tập dữ liệu NSL-KDD đã xử lý (bao gồm cả tập dữ liệu *bình thường* đã huấn luyện và tập dữ liệu *bất thường* chưa huấn luyện) vào mô hình để nhận được kết quả tái tạo. Sau đó, chúng tôi tính độ lỗi tái tạo và đem vào 3 phương pháp phân lớp được trình bày ở trên phần 3.2. Kết quả dưới được thực nghiệm trên bộ dữ liệu KDDTest+.
Độ chính xác của ba phương pháp được trình bày trong bảng 1.

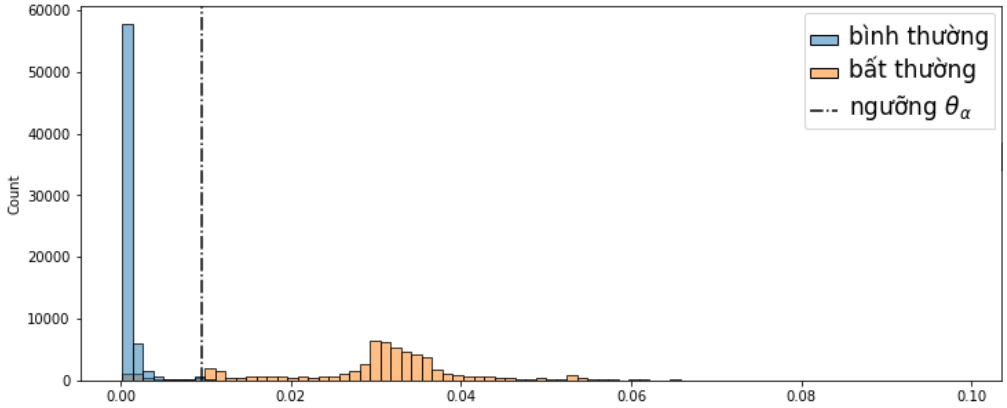
Bảng 1: Độ chính xác của các phương pháp phân lớp được sử dụng trong bài

Phương pháp	**Độ chính xác**
Phân ngưỡng theo θ_α	94.26%
Phân ngưỡng theo Bayes	96.29%
Phân ngưỡng theo Neyman-Pearson	96.39%
Trung bình (cả 3) $\pm$ độ lệch chuẩn (cả 3)	$96.65 \pm 0.98\%$

Theo như hình 9, cả 3 phương pháp phân lớp đều có số lượng FN thấp và FP cao hơn, điều này đúng với tiêu chí "thà phát hiện nhầm còn hơn bỏ sót" chúng tôi đã đề cập. Tỉ lệ dự đoán sai ở hình 10 của các phương pháp được trình bày lần lượt là: ngưỡng cài đặt trước (FNR + FPR = 5.23%), kiểm định giả thuyết Bayes (FNR + FPR = 6.24%), và kiểm định giả thuyết Neyman-Pearson (FNR + FPR = 6.34%).

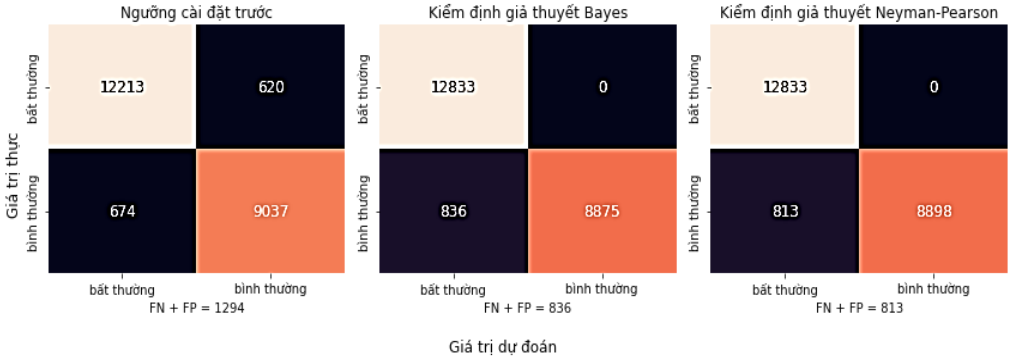


Hình 8: Mô tả AUC

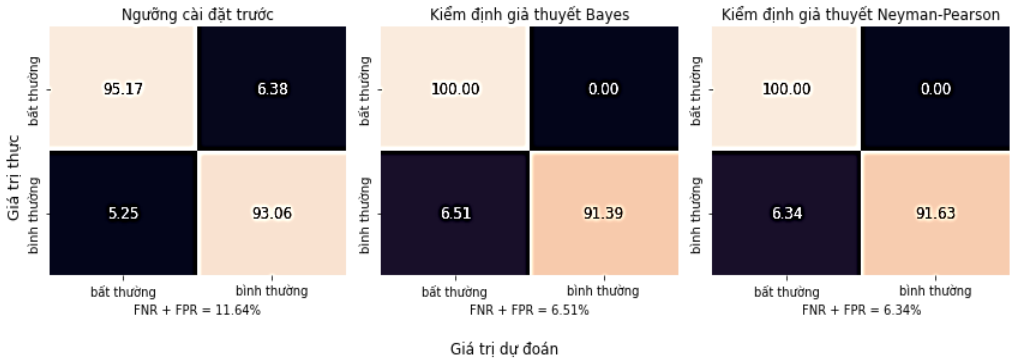


Hình 11: Biểu đồ phân phối dữ liệu tái cấu trúc và vị trí của ngưỡng

Đối với phương pháp sử dụng ngưỡng θ_α, sử dụng trực tiếp lỗi tái tạo để so khớp, nên có thể hình dung đơn giản thông qua hình 11. Ngưỡng màu đen sẽ phân chia thành hai lớp dữ liệu, nếu lỗi tái tạo nhỏ hơn hoặc bằng ngưỡng thì sẽ được phân là bình thường, ngược lại sẽ là bất thường.



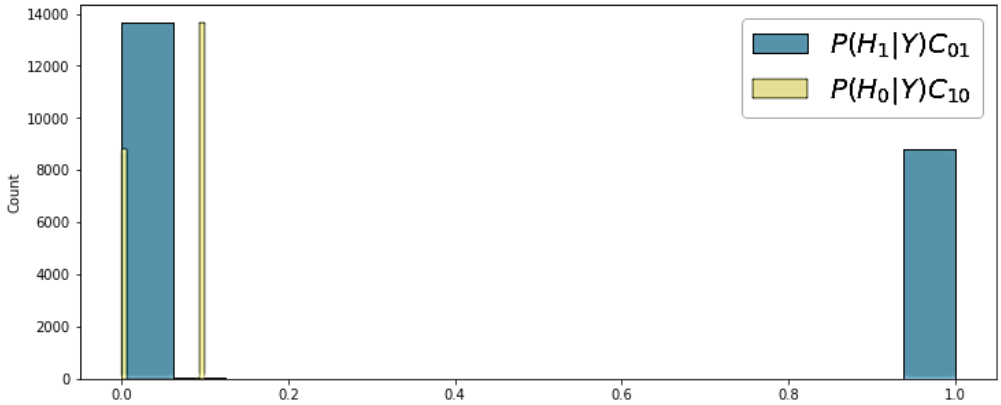
Hình 9: Ma trận nhầm lẫn của ba phương pháp phân loại



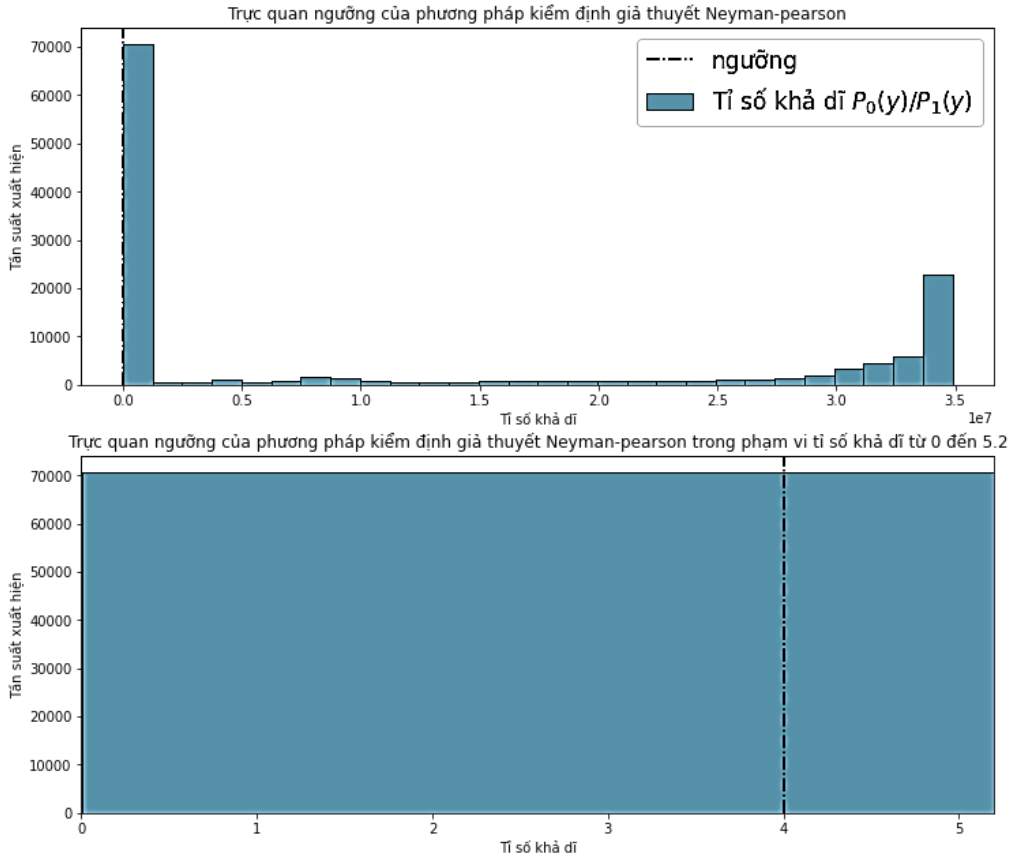
Hình 10: Ma trận nhầm lẫn đã chuẩn hóa của ba phương pháp phân loại

Kiểm định giả thuyết Bayes không trực tiếp sử dụng lỗi tái tạo trên để so khớp, thay vào đó, phương pháp sẽ tính toán và so sánh hai đại lượng: $P(H_1|Y)C_{01}$ và $P(H_0|Y)C_{10}$. Nếu $P(H_1|Y)C_{01} \geq P(H_0|Y)C_{10}$ thì sẽ được coi là bình thường, ngược lại thì bất thường. Hình 11 mô tả các đại lượng này.

Kiểm định giả thuyết Neyman-Pearson cũng giống như kiểm định giả thuyết Bayes, cũng không sử dụng trực tiếp lỗi tái tạo mà sử dụng ngưỡng γ đã thử nghiệm và so sánh trên đại lượng $L = P_0(y)/P_1(y)$. Dữ liệu sẽ được phân là bình thường khi nằm bên trái của ngưỡng, ngược lại sẽ là bất thường. Hình 13 minh họa tỉ số khả dĩ L và ngưỡng.



Hình 12: Biểu đồ thể hiện phân phối của hai đại lượng trong kiểm định giả thuyết Bayes



Hình 13: Mô tả phân phối đại lượng L và vị trí của ngưỡng trong kiểm định giả thuyết Neyman-Pearson. Hình ở dưới mô tả kỹ hơn về vị trí của ngưỡng trên trục tọa độ ngắn hơn.

Ngoài việc tự so sánh kết quả ở trên, chúng tôi còn so sánh kết quả với một vài phương pháp khác có cùng bộ dữ liệu NSL-KDD, thực nghiệm trên KDDTest+. Kết quả được trình bày ở bảng dưới đây, đơn vị (%).

Bảng 2: So sánh 3 phương pháp của chúng tôi với các phương pháp khác

Phương pháp	**AUC**	**Độ chính xác**	**F1-Score**
Jorge Meira và cộng sự [19]			
Isolation Forest	81.71	-	53

K-Means	84.76	-	55
1-Nearest Neighbor	84.85	-	60
Bộ tự mã hóa	83.65	-	63
Scaled Convex Hull	85.30	-	66
Support Vector Machines	83.14	-	65
Wen Xu và cộng sự [20]			
Bộ tự mã hóa 5 lớp + xử lý dữ liệu ngoại lai	-	90.61	92.26
Cosimo Ieracitano và cộng sự [4]			
Bộ tự mã hóa phân loại nhị phân	84.21	-	81.98
Của chúng tôi			
Bộ tự mã hóa + phân ngưỡng theo θ_α	**94.11**	**94.26**	**93.31**
Bộ tự mã hóa + kiểm định giả thuyết Bayes	**95.69**	**96.29**	**95.50**
Bộ tự mã hóa + kiểm định giả thuyết Neyman-Pearson	**96.81**	**96.39**	**95.63**

5. KẾT LUẬN VÀ HƯỚNG NGHIÊN CỨU TƯƠNG LAI

Trong bài báo này, chúng tôi đã đề cập đến bộ tự mã hóa khử nhiễu và sử dụng kết hợp với 3 phương pháp phân lớp: dùng một ngưỡng θ_α, dùng kiểm định giả thuyết Bayes và dùng kiểm định giả thuyết Neyman-Pearson. Sau khi thử nghiệm, thu được kết quả độ chính xác trung bình và độ lệch trên 3 phương pháp là $96.65 \pm 0.98\%$. Trong tương lai, chúng tôi sẽ tập trung vào việc cải tiến hơn nữa bằng cách sử dụng, so sánh kết quả của thuật toán trong bài toán phân lớp đa lớp. Bên cạnh đó, tích hợp các kỹ thuật mới vào IDS, giúp cho hệ thống này không chỉ dừng ở mức dự đoán tấn công mà thậm chí còn có thể phát hiện ngay lập tức và phản ứng lại các tấn công đang diễn ra tại thời điểm đó.

TÀI LIỆU THAM KHẢO

[1] P. Diaz-Gomez and D. F. Hougen, "Improved Off-Line Intrusion Detection Using a Genetic Algorithm," in Proceedings of the Seventh International Conference on Enterprise Information Systems, 2005, pp. 66-73.
[2] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep Learning Approach Combining Sparse Autoencoder With SVM for Network Intrusion Detection," IEEE Access, vol. 6, pp. 52843-52856, 2018.
[3] T. Vaiyapuri and A. Binbusayyis, "Application of deep autoencoder as an one-class classifier for unsupervised network intrusion detection: a comparative evaluation," PeerJ Computer Science, vol. 6, no. 1, p. e327, 2020.
[4] C. Ieracitano, A. Adeel, F. C. Morabito, and A. Hussain, "A Novel Statistical Analysis and Autoencoder Driven Intelligent Intrusion Detection Approach," Neurocomputing, vol. 387, pp. 51-62, 2020.
[5] Y. Yu, J. Long, and Z. Cai, "Network Intrusion Detection through Stacking Dilated Convolutional Autoencoders," Security and Communication Networks, vol. 2017, pp. 1-12, 2017.
[6] Y. Song, S. Hyun, and Y.-G. Cheong, "Analysis of Autoencoders for Network Intrusion Detection," in National Library of Medicine, vol. 21, no. 13, pp. 4294, 2021.
[7] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupalula, "Autoencoder-based feature learning for cyber security applications," in Proc. 2017 International Joint Conference on Neural Networks (IJCNN), pp. 3854-3861, 2017.
[8] Z. Wen and P. He, "Network security situation abnormal detection method based on hypothesis test," in Journal of Computational Methods in Sciences and Engineering, vol. 16, pp. 505-518, 2016.
[9] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised Detection of Anomalous Sound Based on Deep Learning and the Neyman–Pearson Lemma," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 27, no. 1, pp. 212-224, 2019.
[10] "Autoencoder," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Autoencoder. [Accessed: Nov. 14, 2021].
[11] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in 2018 20th International Conference on Advanced Communication Technology (ICACT), pp. 178-183, 2018.
[12] H. Choi, M. Kim, G. Lee, and W. Kim, "Unsupervised learning approach for network intrusion detection system using autoencoders," The Journal of Supercomputing, vol. 75, no. 9, pp. 5597–5621, 2019.

[13] T. Alpcan and T. Basar, Network Security: A Decision and Game-Theoretic Approach. Cambridge University Press, 2010.
[14] H. Pishro-Nik, Introduction to Probability, Statistics, and Random Processes. Kappa Research, LLC, 2014.
[15] T. Alpcan and T. Basar, "Network Security A Decision and Game-Theoretic Approach," 2010.
[16] R. Lippmann, "Lincoln Laboratory Masssachusetts Institute of Technology," Feb. 15, 2019. [Online]. Available: https://archive.ll.mit.edu/ideval/files/1999_DARPA_EvaulationSumPlans.pdf. [Accessed: Oct. 2, 2021].
[17] "NSL-KDD: Góc nhìn chi tiết về tập dữ liệu huấn luyện cho các IDS," [Online]. Available: https://inseclab.uit.edu.vn/nsl-kdd-goc-nhin-chi-tiet-ve-tap-du-lieu-huan-luyen-cho-cac-ids/. [Accessed: Feb. 25, 2021].
[18] "Evaluation," [Online]. Available: https://machinelearningcoban.com/2017/08/31/evaluation/. [Accessed: Dec. 24, 2021].
[19] J. Meira, R. Andrade, I. Praça, J. Carneiro, V. Bolón-Canedo, A. Alonso-Betanzos, and G. Marreiros, "Performance evaluation of unsupervised techniques in cyber-attack anomaly detection," Journal of Ambient Intelligence and Humanized Computing, vol. 11, 2020.
[20] W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and S. Fariza, "Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset," IEEE Access, vol. 9, pp. 140136-140146, 2021.

USING BAYESIAN AND NEYMAN-PEARSON HYPOTHESIS TESTING FOR AUTOENCODER TO DETECT ANOMALIES IN NETWORK SECURITY

NGUYEN VAN ANH TUAN *, DINH HOANG HAI DANG, TRAN NAM BA,
NGUYEN THI THANH HOA, TRINH THI BAO BAO, PHAN LE HOANG VIET,
NGUYEN CHI KIEN, NGUYEN HUU TINH
Faculty of Information Technology, Industrial University of Ho Chi Minh City
**Corressponding author: nvatuan3@gmail.com*

Abstract. An autoencoder is an unsupervised learning model whose parameters are finetuned so that the output vector is as close as possible to the input vector. In this paper, we use the autoencoder model to detect abnormal (attack) connections from normal connections on the Internet. The reconstruction error of the autoencoder will be used to classify connections into two classes: normal connections and abnormal connections. We present three methods to classify the connections: using a given threshold, Bayesian hypothesis testing, and Neyman-Pearson hypothesis testing. On the NSL KDD dataset, the average accuracy achieved by the three methods was $96.65 \pm 0.98\%$.
Keywords. Autoencoder, Bayesian hypothesis testing, Neyman-Pearson hypothesis testing, anomaly detection.

Ngày gửi bài: 16/03/2022
Ngày chấp nhận đăng: 16/08/2022