HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Trần Huỳnh Tiến

ỨNG DỤNG REPRESENTATION LEARNING PHÁT HIỆN TẤN CÔNG PHISHING

Chuyên ngành: Hệ thống thông tin

Mã số: 8.48.01.04

TÓM TẮT LUẬN VĂN THẠC SĨ

TP.HCM - NĂM 2023

Luận văn được hoàn thành tại:

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Người hướng dẫn khoa học: **TS. NGUYỄN HỒNG SƠN**

Phản biện 1: ----------

Phản biện 2: ----------

Luận văn sẽ được bảo vệ trước Hội đồng chấm luận văn thạc sĩ tại Học viện Công nghệ Bưu chính Viễn thông

Vào lúc: giờ ngày tháng năm 2023.

Có thể tìm hiểu luận văn tại:

- Thư viện của Học viện Công nghệ Bưu chính Viễn thông.

PHẦN MỞ ĐẦU

1. Lý do chọn đề tài

Tấn công lừa đảo (Phishing) là hình thức tấn công phi kỹ thuật được tội phạm mạng sử dụng nhiều nhằm đánh cắp dữ liệu bí mật từ máy tính hay một mạng máy tính của người dùng, sau đó sử dụng dữ liệu cho nhiều mục đích khác nhau, như lấy cắp tiền của nạn nhân hoặc bán lại dữ liệu đã đánh cắp.

Sự phát triển của trí tuệ nhân tạo, máy học trong những năm gần đây rất có tiềm năng áp dụng để phát hiện tấn công Phishing với độ chính xác cao. Trong đó mô hình dựa vào máy học có thể phát huy nhiều ưu điểm cho bài toán này. Xuất phát từ thực tế đó đề cương luận văn tập trung nghiên cứu:

"Ứng dụng representation learning phát hiện tấn công Phishing"

2. Tổng quan về vấn đề nghiên cứu

Nghiên cứu các tài liệu liên quan đến đề tài, học viên nhận thấy độ chính xác và thời gian phát hiện tấn công giả mạo là hai yếu tố quan trong. Trong đề tài này sẽ tập trung

vào hai yếu tố trên để tăng hiệu quả khả năng phát hiện xâm nhập với thời gian phù hợp nhất.

3. Mục đích nghiên cứu

Mục tiêu chính: Xây dựng mô hình máy học sử dụng phương pháp representation learning để phát hiện tấn công phishing nhằm nâng cao độ chính xác của phát hiện.

4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu chính là tấn công Phishing và phương pháp representation learning nghiên cứu các mô hình dự báo áp dụng vào phương pháp representation learning.

5. Phạm vi nghiên cứu

Xây dựng mô hình mô phỏng máy học, sử dụng phương pháp để phát hiện tấn công Phishing.

6. Phương pháp nghiên cứu

Phương pháp luận: Dựa trên cơ sở là lý thuyết về phương pháp RL; Dự kiến dùng mô hình RL học viên áp dụng các phương pháp Deep Learning và HTML Analysis

Phương pháp đánh giá dựa trên cơ sở toán học: Trên cơ sở các lý thuyết về phương pháp RL.

Phương pháp đánh giá bằng mô hình mô phỏng thực nghiệm: Xây dựng mô hình mô phỏng và thực nghiệm đề hoàn thành đề xuất

7. Bố cục luận văn

Chương 1: Tổng quan tấn công phishing và representation learning

Chương 2: Xây dựng mô hình phát hiện tấn công phishing

Chương 3: Thí nghiệm và đánh giá

CHƯƠNG 1: TỔNG QUAN TẤN CÔNG PHISHING VÀ REPRESENTATION LEARNING

1.1. Tổng quan về tấn công phishing

Phishing là một trong những loại tấn công mạng nguy hiểm do các tội phạm mạng gây ra bằng cách tạo ra các thông tin giả mạo từ các website, cơ sở, doanh nghiệp uy tín nhằm lừa đảo và chiếm đoạt thông tin của người dùngTổng quan về cân bằng tải trong điện toán đám mây

1.2. Các phương pháp phòng chống và phát hiện Phishing trên mạng

Tấn công Phishing luôn tiềm ẩn và khó nhận biết vì mức độ tinh vi của nó với bất kì cá nhân hoặc tổ chức nào, vì vậy các cá nhân hoặc tổ chức cần nâng cao cảnh giác đối với các loại tài khoản cũng như thông tin cá nhân của mình. Một số cách phòng chống tấn công Phishing được đề xuất như sau:

Cấu hình tài khoản: các tổ chức nên cấu hình các loại tài khoản của nhân viên theo nguyên tắc giảm thiểu tối

đa các loại đặc quyền, chỉ cấp các quyền cần thiết cho nhân viên

Tập huấn cho nhân viên: Các nhân viên trong một tổ chức cần được tập huấn kiến thức cũng như nhận biết được những lúc hệ thống có các hoạt động bất thường

Kiểm tra các dấu hiệu của Phishing: nâng cao cảnh giác với một số email đến từ nước ngoài, có nội dung không hoàn chỉnh (lỗi chính tả, sai dấu chấm câu,…).

Báo cáo lại tất cả các cuộc tấn công: báo cáo lại với cấp trên để được hỗ trợ kịp thời, tránh những rủi ro đáng tiếc xảy ra

Kiểm tra dấu vết thông tin cá nhân: cần ý thức đến việc chia sẻ thông tin nhạy cảm về cơ quan, tổ chức hoặc thông tin cá nhân trên các trang mạng xã hội để tránh các cuộc tấn công có thể xảy đến.

1.3. Tổng quan về về representation learning

Representation learning là tập hợp các kỹ thuật cho phép một hệ thống tự động khám phá các biểu diễn cần thiết để phát hiện hoặc phân loại đặc trưng từ bộ dữ liệu thô.

Supervised representation learning: học các biểu diễn về nhiệm vụ A bằng cách sử dụng dữ liệu được chú thích và được sử dụng để giải quyết nhiệm vụ B.

Unsupervised representation learning: học các biểu diễn về một nhiệm vụ theo cách không được giám sát (dữ liệu không có nhãn.

Các kĩ thuật Representation Learning lần đầu tiên được phát triển để phục vụ cho quá trình xử lí ngôn ngữ tự nhiên, tuy nhiên chúng đã được mở rộng sang kiểu xử lí dữ liệu khác như là hình ảnh, video và hệ thống mạng

1.4. Một số đặc điểm nổi bật của representation learning.

Ưu tiên cho RL trong AI

Smoothness: giả sử hàm được học f là s.t. $x \approx y$ thường ngụ ý $f(x) \approx f(y)$

Nhiều yếu tố giải thích: phân phối tạo dữ liệu được tạo ra bởi các yếu tố cơ bản khác nhau và phần lớn những gì người ta tìm hiểu về một yếu tố sẽ khái quát trong nhiều cấu hình của các yếu tố khác.

Một tổ chức có thứ bậc của các yếu tố giải thích: các khái niệm hữu ích để mô tả thế giới xung quanh có thể được

định nghĩa theo các khái niệm khác, trong một hệ thống thứ bậc, với các khái niệm trừu tượng hơn trong hệ thống thứ bậc, được định nghĩa theo các khái niệm ít trừu tượng hơn.

Học bán giám sát: với đầu vào X và mục tiêu Y để dự đoán, một tập hợp con của các yếu tố giải thích phân phối của X giải thích phần lớn Y, cho X. Do đó, các biểu diễn hữu ích cho P (X) có xu hướng hữu ích khi học P (Y | X), cho phép chia sẻ sức mạnh thống kê giữa các nhiệm vụ học tập được giám sát và không giám sát.

Các yếu tố được chia sẻ giữa các nhiệm vụ: với nhiều Y quan tâm hoặc nhiều nhiệm vụ học tập nói chung, các nhiệm vụ (ví dụ: tương ứng với P (Y | X, nhiệm vụ)) được giải thích bằng các yếu tố được chia sẻ với các nhiệm vụ khác, cho phép chia sẻ các điểm mạnh thống kê qua các nhiệm vụ.

Manifolds: khối lượng xác suất tập trung gần các vùng có kích thước nhỏ hơn nhiều so với không gian ban đầu nơi dữ liệu tồn tại.

Phân cụm tự nhiên: các giá trị khác nhau của các biến phân loại như các lớp đối tượng được liên kết với các đa tạp riêng biệt.

Tính nhất quán theo thời gian và không gian: các quan sát liên tiếp (từ một trường hợp) hoặc các quan sát gần nhau về mặt không gian có xu hướng được liên kết với cùng một giá trị của các khái niệm phân loại có liên quan, hoặc dẫn đến một chuyển động nhỏ trên bề mặt của đa tạp mật độ cao.

Độ thưa thớt: đối với bất kỳ quan sát x đã cho nào, chỉ một phần nhỏ các yếu tố có thể là có liên quan.

Tính đơn giản của các yếu tố phụ thuộc: trong các biểu diễn cấp cao, các yếu tố có liên quan với nhau thông qua các phụ thuộc tuyến tính, đơn giản.

Các yếu tố bất đồng của sự thay đổi

Các yếu tố giải thích khác nhau của dữ liệu có xu hướng thay đổi độc lập với nhau trong phân phối đầu vào và chỉ một số yếu tố tại thời điểm có xu hướng thay đổi khi người ta xem xét một chuỗi các đầu vào liên tiếp trong thế giới thực.

1.5. Mạng Nơ-ron và Deep learning

1.5.1. Mạng Nơ-ron

Neural network là một mạng lưới thần kinh được tạo thành từ các nút xử lý được kết nối dày đặc, tương tự như các tế bào thần kinh trong não.

1.5.2. Deep learning

Deep learning (DL) hay học sâu là một tập con của học máy (ML), về cơ bản là một mạng nơ-ron có ba lớp trở lên. DL thúc đẩy nhiều ứng dụng và dịch vụ trí tuệ nhân tạo (AI) nhằm cải thiện tự động hóa, thực hiện các tác vụ phân tích và vật lý mà không cần sự can thiệp của con người.

1.6. Các công trình nghiên cứu liên quan ở trong nước

Vào năm 2014, tác giả Phạm Tuấn Anh cùng các cộng sự của mình đã đề xuất giải pháp chống tấn công Phishing bằng Genetic Programming (GP) giải pháp đã chứng minh tính hiệu quả cao và được nhóm tác giả cho là giải pháp tốt nhất cho việc phát hiện các cuộc tấn công lừa đảo.

Tác giả Le Dang Nguyen, Đại học Hải Phòng, năm 2014 cùng các cộng sự của mình nghiên cứu và đề xuất các giải pháp để phát hiện các trang web lừa đảo, giả mạo đựa

trên cấu trúc của cây DOM (DOM-Tree) và thuật toán Graph Matching.

Vào năm 2018, Do Xuan Cho cùng các cộng sự của mình đã thực hiện nghiên cứu về hệ thống phòng chống tấn công Phishing qua email cho người Việt Nam.

1.7. Các công trình nghiên cứu liên quan trên thế giới

Một số công trình tiêu biểu;

Yoshua Bengio cùng các cộng sự của mình đã thực hiện bài đánh giá và giới thiệu về thuật toán vô cùng mạnh mẽ trong lĩnh vực ML và DL là RL.

"An overview on data representation learning: From traditional feature learning to recent deep learning" bài báo này xem xét nghiên cứu về học biểu diễn dữ liệu, bao gồm học tập tính năng truyền thống và học tập sâu.

"RLOSD: Representation Learning based Opinion Spam Detection" bài báo này đề xuất một phương pháp dựa trên cây quyết định để tiết lộ các đánh giá lừa đảo từ những người đáng tin cậy..

"A Survey on Representation Learning Efforts in Cybersecurity Domain" bài báo đã thảo luận về các cuộc

tấn công mạng khác nhau và các sáng kiến được thực hiện bởi các khu tổ chức quốc tế.

CHƯƠNG 2: XÂY DỰNG MÔ HÌNH PHÁT HIỆN TẤN CÔNG PHISHING

2.1. Thiết kế mô hình

Trong luận văn này, với tính chất của các url của cách tấn công phishing, sử dụng tokenization để chuyển thành ma trận số dựa vào xử lý ngôn ngữ tự nhiên các url. Từ đó, chuyển ma trận url này thành ma trận hình ảnh grayscale và áp dụng ResNet18 để training và xây dựng mô hình nhận diện Phishing.
Với ý tưởng này, luận văn đề xuất như xây dựng mô hình như sau:
(1) Url → Tokenization → Text_to_matrix → numpy Matrix

(2) Numpy Matrix → convert to Image Matrix (Gray scale) → Array of images

(3) Array of Images → training with ResNet → Model

Trong thử nghiệm, các tính năng của trang web được chuyển đổi thành các vectơ đặc trưng và được sử dụng làm đầu vào cho các mô hình DL.

- Với ý tưởng phát triên mô hình làm việc như bộ phân loại. Mô hình nhận dữ liệu chuỗi URL từ đó, chỉ ra chính xác dữ liệu tấn công phishing hay không.

Một trong các mô hình representation learning đơn giản là RestNet18, và cách biến đổi URL thành ma trận

thông qua Tokenizer của TensorFlow. Từ 2 kỹ thuật phổ biến này, luận văn tích hợp và xây dựng mô hình phát hiện tấn công Phishing.

2.2. Bộ dữ liệu của bài toán

Luận văn sử dụng bộ dữ liệu Web page Phishing Detection. Bộ dữ liệu bao gồm 11,430 dòng và 89 cột cung cấp 11,429 URL với 87 tính năng được trích xuất.

Trong tổng số 89 trường dữ liệu, trường url và status mang giá trị chuỗi, các trường còn lại mang kiểu dữ liệu số nguyên (chiếm đa số) hoặc số thực.

Với mỗi thuộc tính, quan sát các đặc điểm như miêu tả về tổng số dữ liệu hợp lệ, giá trị trung bình, độ lệch chuẩn, giá trị nhỏ nhất, từ giá trị của dữ liệu trở xuống chiếm 25%, 50% và 75%, giá trị lớn nhất.

2.3 Phương pháp đánh giá

Độ chính xác (hay còn gọi là accuracy) sẽ được sử dụng trong trường hợp này đo bằng công thức như sau:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Đối với phân loại nhị phân, độ chính xác cũng có thể được tính theo mặt tích cực (Positive) và tiêu cực (Negative) với công thức như sau:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2.4. Hiện thực mô hình

Sử dụng máy chủ đám mây cho phép tận dụng sức mạnh của phần cứng của Google Colab để luyện mô hình.

Ngôn ngữ lập trình được viết bằng mã Python với sự trợ giúp của gói TensorFlow.

2.4.1. Xử lý các URL

Sử dụng Keras Tokenizer để chuyển bộ dữ liệu url thành tập các ma trận. Sau khi xử lý, các ma trận có kích thước là (37x37). Sau đó chuyển sang hình ảnh gray-scale.

2.4.2. Xây dựng mô hình ResNet18

Để xác định tiền xử lý cho dữ liệu ảnh, chúng ta sẽ phải lật ngang ngẫu nhiên, xoay, chuẩn hóa, v.v. Sau đó, thay đổi kích thước hình ảnh phải là (n * n) vì Resnet chấp nhận kích thước hình ảnh đầu vào là (n * n). Chia tập dữ liệu thành train và test với tỉ lệ 8:2. Tạo ra các trọng số được train trước cho mô hình resnet18 và thay đổi các lớp của nó và phân loại các lớp cụ thể, trong khi Resnet-18 được đào tạo trên nhiều lớp. Xây dựng mô hình sử dụng chức năng tối ưu hóa và mất mát: trình tối ưu hóa SGD và mất mất mát

Cross-Entropy. Xây dựng mô hình, huấn luyện với 150 vòng trở lên.

CHƯƠNG 3
THI NGHIỆM VÀ ĐÁNH GIÁ

3.1. Các trường hợp thí nghiệm

Sử dụng máy chủ đám mây cho phép tận dụng sức mạnh của phần cứng của Google Colab để luyện mô hình.

Bộ dữ liệu bao gồm 11,430 dòng và 89 cột cung cấp 11,429 URL với 87 tính năng được trích xuất.

Trong quá trình thí nghiệm huấn luyện xây dựng mô hình, để tìm ra mô hình phù hợp với bộ dữ liệu, luận văn đề xuất 4 trường hợp chuyển dữ liệu URL dạng text sang dữ liệu URL dạng numpy matrix với 4 kích thước từ 37x37 sang (75x75), (100x100), (192x192), (224x224) từ đó convert thành ảnh grayscale. Sau đó chạy huấn luyện với tỷ lệ tập train / tập test là 80 / 20. Số epoch chạy cho trường hợp này là 150 epoches.

3.2. Luyện và kiểm thử mô hình

Về cài đặt, cả 4 trường hợp đều cài đặt như nhau:

```python
#Now, we need to start training, if the above steps work fine, then you can easily start training with the below code
num_epochs = 150   #(set no of epochs)
start_time = time.time() #(for showing time)
for epoch in range(num_epochs): #(loop for every epoch)
    print("Epoch {} running".format(epoch)) #(printing message)
    """ Training Phase """
    model.train()    #(training model)
    running_loss = 0.   #(set loss 0)
    running_corrects = 0
    # load a batch data of images
    for i, (inputs, labels) in enumerate(train_dataloader):
        inputs = inputs.to(device)
        labels = labels.to(device)
        # forward inputs and get output
        optimizer.zero_grad()
        outputs = model(inputs)
        _, preds = torch.max(outputs, 1)
        loss = criterion(outputs, labels)
        # get loss value and update the network weights
        loss.backward()
        optimizer.step()
        running_loss += loss.item() * inputs.size(0)
        running_corrects += torch.sum(preds == labels.data)
    epoch_loss = running_loss / len(train_dataset)
    epoch_acc = running_corrects / len(train_dataset) * 100.
    print('[Train #{}] Loss: {:.4f} Acc: {:.4f}% Time: {:.4f}s'.format(epoch, epoch_loss, epoch_acc, time.time() -start_time))

    """ Testing Phase """
    model.eval()
    with torch.no_grad():
        running_loss = 0.
        running_corrects = 0
        for inputs, labels in test_dataloader:
            inputs = inputs.to(device)
            labels = labels.to(device)
            outputs = model(inputs)
            _, preds = torch.max(outputs, 1)
            loss = criterion(outputs, labels)
            running_loss += loss.item() * inputs.size(0)
            running_corrects += torch.sum(preds == labels.data)
        epoch_loss = running_loss / len(test_dataset)
        epoch_acc = running_corrects / len(test_dataset) * 100.
        print('[Test #{}] Loss: {:.4f} Acc: {:.4f}% Time: {:.4f}s'.format(epoch, epoch_loss, epoch_acc, time.time()- start_time))
```

Kết quả thu được sau khi chạy trường hợp 1

```
Epoch 146 running
[Train #146] Loss: 0.5116 Acc: 73.2065% Time: 2930.2671s
[Test #146] Loss: 0.5900 Acc: 70.7787% Time: 2933.1114s
Epoch 147 running
[Train #147] Loss: 0.5108 Acc: 73.6002% Time: 2949.5793s
[Test #147] Loss: 0.5741 Acc: 71.7848% Time: 2952.4797s
Epoch 148 running
[Train #148] Loss: 0.5110 Acc: 73.3158% Time: 2969.2179s
[Test #148] Loss: 0.6412 Acc: 71.5223% Time: 2972.1125s
Epoch 149 running
[Train #149] Loss: 0.5129 Acc: 73.6439% Time: 2990.0927s
[Test #149] Loss: 0.6498 Acc: 68.8539% Time: 2993.0100s
```

Kết quả thu được sau khi chạy trường hợp 2

```
Epoch 145 running
[Train #145] Loss: 0.1528 Acc: 93.4165% Time: 5267.7632s
[Test #145] Loss: 1.0975 Acc: 67.1479% Time: 5272.7419s
Epoch 146 running
[Train #146] Loss: 0.1621 Acc: 92.8696% Time: 5303.2566s
[Test #146] Loss: 1.1158 Acc: 65.9668% Time: 5308.1906s
Epoch 147 running
[Train #147] Loss: 0.1567 Acc: 93.2415% Time: 5338.7614s
[Test #147] Loss: 1.1218 Acc: 66.0980% Time: 5343.6832s
Epoch 148 running
[Train #148] Loss: 0.1576 Acc: 93.3618% Time: 5374.2988s
[Test #148] Loss: 1.1257 Acc: 67.5853% Time: 5379.2483s
Epoch 149 running
[Train #149] Loss: 0.1575 Acc: 93.2415% Time: 5409.8471s
[Test #149] Loss: 1.0184 Acc: 68.8539% Time: 5414.7810s
```

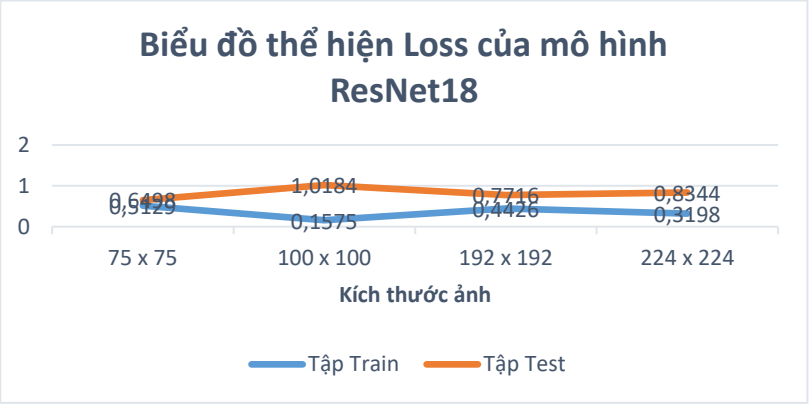
Kết quả thu được sau khi chạy trường hợp 3

```
Epoch 144 running
[Train #144] Loss: 0.1558 Acc: 93.3399% Time: 5230.3393s
[Test #144] Loss: 1.0467 Acc: 67.9353% Time: 5237.1176s
Epoch 145 running
[Train #145] Loss: 0.1528 Acc: 93.4165% Time: 5267.7632s
[Test #145] Loss: 1.0975 Acc: 67.1479% Time: 5272.7419s
Epoch 146 running
[Train #146] Loss: 0.1621 Acc: 92.8696% Time: 5303.2566s
[Test #146] Loss: 1.1158 Acc: 65.9668% Time: 5308.1906s
Epoch 147 running
[Train #147] Loss: 0.1567 Acc: 93.2415% Time: 5338.7614s
[Test #147] Loss: 1.1218 Acc: 66.0980% Time: 5343.6832s
Epoch 148 running
[Train #148] Loss: 0.1576 Acc: 93.3618% Time: 5374.2988s
[Test #148] Loss: 1.1257 Acc: 67.5853% Time: 5379.2483s
Epoch 149 running
[Train #149] Loss: 0.1575 Acc: 93.2415% Time: 5409.8471s
[Test #149] Loss: 1.0184 Acc: 68.8539% Time: 5414.7810s
```

Kết quả thu được sau khi chạy trường hợp 4

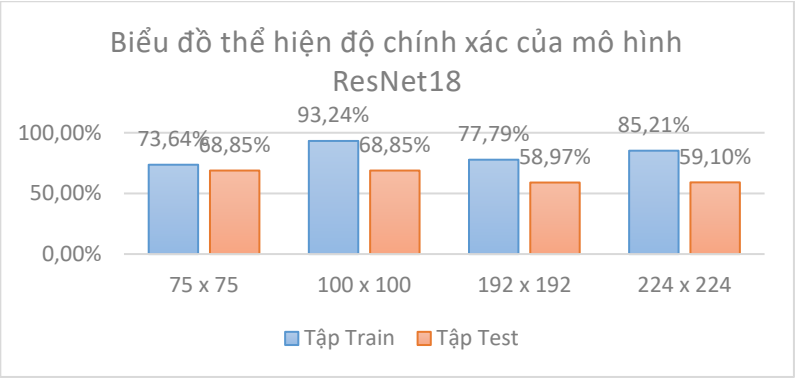
```
Epoch 145 running
[Train #145] Loss: 0.3310 Acc: 84.4926% Time: 6153.5694s
[Test #145] Loss: 0.8597 Acc: 58.7927% Time: 6158.8872s
Epoch 146 running
[Train #146] Loss: 0.3325 Acc: 84.3066% Time: 6195.5201s
[Test #146] Loss: 0.8393 Acc: 61.5048% Time: 6200.9193s
Epoch 147 running
[Train #147] Loss: 0.3255 Acc: 84.8535% Time: 6237.6464s
[Test #147] Loss: 0.8769 Acc: 61.1549% Time: 6243.0402s
Epoch 148 running
[Train #148] Loss: 0.3226 Acc: 85.3565% Time: 6279.5259s
[Test #148] Loss: 0.9210 Acc: 59.7550% Time: 6284.9492s
Epoch 149 running
[Train #149] Loss: 0.3198 Acc: 85.2144% Time: 6321.5848s
[Test #149] Loss: 0.8344 Acc: 59.0989% Time: 6328.7170s
```

3.3. Kết quả và nhận xét



Hình 3.1: Biểu đồ thể hiện Loss của mô hình ResNet18 với 4 trường hợp

Độ mất mát của tập Train luôn thấp hơn tập test.



Hình 3.2: Biểu đồ thể hiện Accuracy của mô hình ResNet18 với 4 trường hợp

Ở trường hợp kích thước ảnh (100x100) thì tập train đạt độ chính xác cao nhất 93.24%. Tuy nhiên, ở trường hợp kích thước ảnh (75x75), (192x192), (224x224) thì tập test và tập train chênh lệch không cao, độ chính xác dao động từ 58% tới 85%. Vì vậy, kích thước ảnh thay đổi dẫn đến chất lượng mô hình thay đổi theo. Với kết quả như trên rõ ràng với các kích thước ảnh (100x100) đạt độ chính xác cao vì đây là kích thước ảnh thuận lợi cho việc học và suy diễn trong mạng CNN.

Việc sử dụng máy chủ đám mây cho phép tận dụng sức mạnh của phần cứng của Google Colab để luyện mô hình chỉ phù hợp với bộ dữ liệu có dung lượng nhỏ nên kết quả độ chính xác chưa cao. Bộ dữ liệu bao gồm 11,430 dòng và 89 cột cung cấp 11,429 URL với 87 tính năng được trích xuất.

KẾT LUẬN

Kết quả nghiên cứu của đề tài

Phát hiện tấn công Phishing đã trở thành một đề tài nghiên cứu và là một bài toán vô cùng quan trọng trong an toàn thông tin. Sự phát triển của trí tuệ nhân tạo trong thời gian gần đây góp phần ngăn chặn, phát hiện tấn công Phishing với độ chính xác cao. Trong đó mô hình representation learning đã phát huy nhiều ưu điểm cho vấn đề này.

Hạn chế luận văn

Luận văn sử dụng bộ dữ liệu công bố, nên cần phải đưa bộ dữ liệu thực tế để tiến hành nghiên cứu và đo đạc.

Chưa cài đặt được trên môi trường mạng thực tế mà dừng lại ở mức thực nghiệm phân tích xây dựng mô hình trên dataset.

Vấn đề kiến nghị và hướng đi tiếp theo của nghiên cứu:

- Xây dựng bộ dữ liệu thực tế với tình hình cyber security ở Việt Nam và thời điểm hiện tại, cập nhật liên tục.

- Cài đặt mô hình và ứng dụng realtime vào mô hình cho việc phát hiện Phishing trên mạng thực.