**ĐẠI HỌC QUỐC GIA TP.HCM**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**MÔN HỌC: AN TOÀN MẠNG**

**BÁO CÁO ĐỒ ÁN CUỐI KÌ**

**Giảng viên hướng dẫn: TS. Phạm Văn Hậu**

**Mã lớp học: NT140.O11.ANTN**

**Tên chủ đề: Malware Traffic Classification Using Domain Adaptation and Ladder Network for Secure Industrial Internet of Things**

**(Phân loại lưu lượng mã độc sử dụng Thích ứng miền và Ladder Network cho an toàn trong IIoT)**

**Sinh viên thực hiện:**

**21522641 – Phạm Ngọc Thơ**

**21522056 – Hà Thị Thu Hiền**

**21521248 – Nguyễn Ngọc Nhung**

**TP.HCM, ngày 30 tháng 12 năm 2023**

**TÓM TẮT NỘI DUNG**

Mục lục

[Mục lục 2](#_Toc154925863)

[I. Giới thiệu 2](#_Toc154925864)

[1. Các khái niệm cơ bản 2](#_Toc154925865)

[2. Ngữ cảnh, các đóng góp chính của bài báo 4](#_Toc154925866)

[3. Mô hình hệ thống 4](#_Toc154925867)

[II. Phương pháp chính được sử dụng trong bài báo 7](#_Toc154925868)

[1. Tiền xử lý dữ liệu 7](#_Toc154925869)

[2. Quá trình đào tạo các model 7](#_Toc154925870)

[III. Kết quả thực nghiệm của bài báo 12](#_Toc154925871)

[1. Môi trường thực nghiệm 12](#_Toc154925872)

[2. Kết quả thực nghiệm 13](#_Toc154925873)

[3. Kết luận của bài báo 15](#_Toc154925874)

[IV. Kết quả thực nghiệm của nhóm 16](#_Toc154925875)

[1. Môi trường thực nghiệm 16](#_Toc154925876)

[2. Xử lý dataset 16](#_Toc154925877)

[3. Kết quả thực nghiệm 23](#_Toc154925878)

[V. Kết luận: 24](#_Toc154925879)

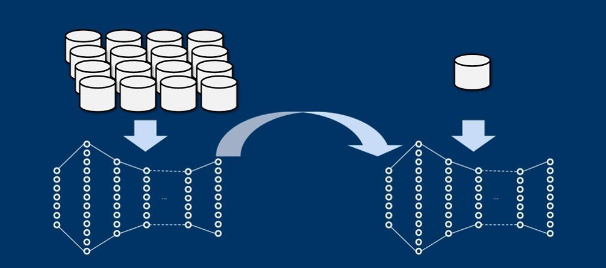
**BÁO CÁO CHI TIẾT**

## Giới thiệu

### Các khái niệm cơ bản

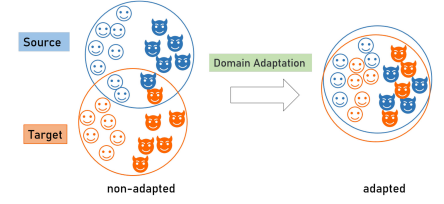
* Transfer learning (học chuyển giao): là một phương pháp trong lĩnh vực machine learning, là việc sử dụng lại pre-trained model cho một vấn đề mới. Tóm lại là sử dụng kiến thức thu được từ nhiệm vụ cũ để xử lý nhiệm vụ mới có liên quan, giúp cải thiện hiệu suất huấn luyện và tăng tốc học. TL được sử dụng trong trường hợp dữ liệu hạn chế hoặc có nhiều nhiệm vụ liên quan với nhau.

Ví dụ: Khi đào tạo bộ phân loại để dự đoán liệu một hình ảnh có chứa thức ăn hay không, ta có thể sử dụng kiến thức mà nó thu được trong quá trình đào tạo để nhận dạng hình ảnh chứa đồ uống.



* Domain adaptation (thích ứng miền): là một nhánh của transfer learning, là khả năng áp dụng thuật toán được đào tạo trong một hoặc nhiều "miền nguồn" sang "miền đích" khác (nhưng có liên quan). Điều khác biệt của DA so với TL là trong DA, miền nguồn và miền đích đều có cùng một *features space* (nhưng phân bổ khác nhau). Mục tiêu của DA là giảm sự khác biệt về phân phối miền giữa miền nguồn và miền đích, từ đó tập trung vào việc cải thiện hiệu suất trên một tập dữ liệu mới mà không cần phải huấn luyện lại mô hình từ đầu. Chủ yếu được sử dụng khi có sự chênh lệch giữa các tập dữ liệu và mô hình cần được điều chỉnh để hoạt động hiệu quả trên tập dữ liệu mới.

Ví dụ: Chuyển giao mô hình từ ngôn ngữ tự nhiên trong tin tức sang ngôn ngữ trong môi trường y tế.

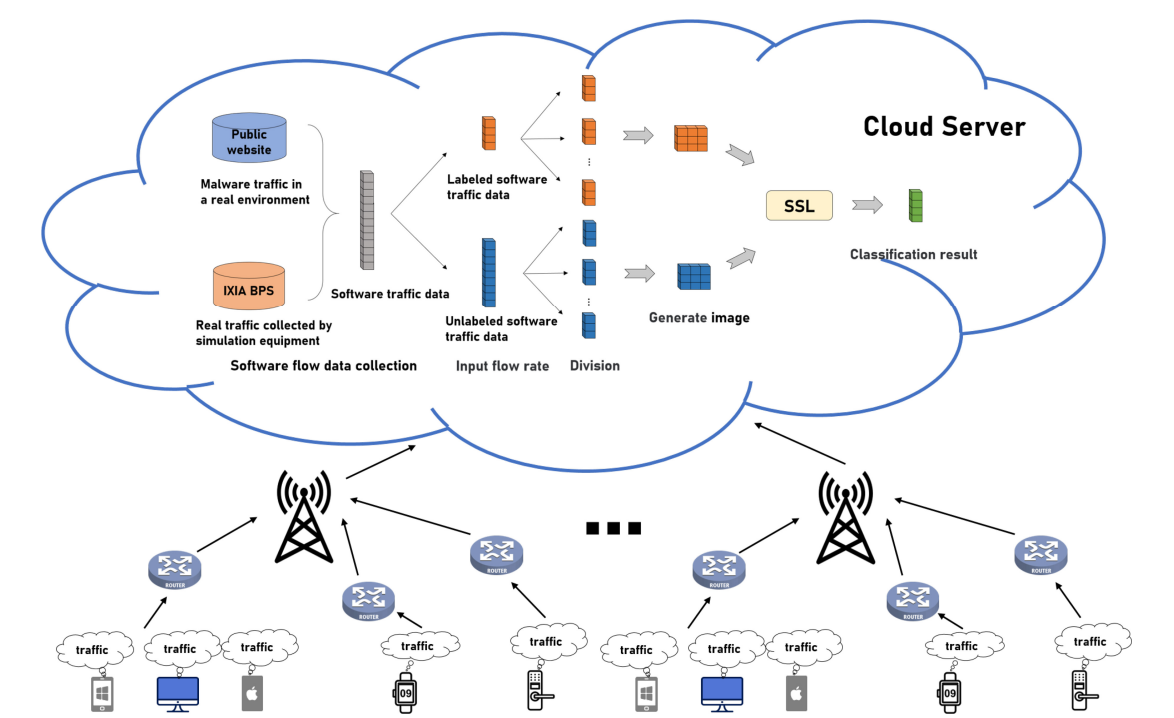


* Ladder Network: là sự kết hợp giữa học tập có giám sát với học tập không giám sát trong mạng lưới thần kinh sâu; thường được áp dụng trong các bài toán không giám sát, nơi mà không có nhãn cho dữ liệu huấn luyện và mô hình cần học cách biểu diễn dữ liệu một cách hiệu quả.
* Semisupervised learning (học bán giám sát): là sự kết hợp giữa học có giám sát và học không giám sát, bằng cách huấn luyện đồng thời một lượng nhỏ dữ liệu được dán nhãn và một lượng lớn dữ liệu không được gắn nhãn để giải quyết vấn đề thiếu mẫu có nhãn trong học có giám sát.
* Các cụm từ viết tắt:
* SSL: semisupervised learning, nghĩa là “học bán giám sát”.
* DA: domain adaptation, “thích ứng miền”.
* TL: transfer learning, “học chuyển giao”.

### Ngữ cảnh, các đóng góp chính của bài báo

* Ngữ cảnh: Các phương pháp phân loại lưu lượng mạng mã độc truyền thống có độ chính xác thấp. Các phương pháp học sâu thì chính xác cao hơn, nhưng trong các kịch bản ứng dụng thực tế, các phương pháp học sâu đòi hỏi một lượng lớn dữ liệu được dán nhãn, tuy nhiên điều này tốn chi phí rất lớn để thực hiện. Do đó, bài báonày đề xuất ba phương pháp dựa trên học bán giám sát (SSL), học chuyển giao (TL) và miền thích ứng (DA) để dù không có nhiều dữ liệu được dán nhãn nhưng vẫn đạt được độ chính xác cao cho mô hình. Trong bài viết này, tác giả đề xuất phương pháp ConvLaddernet dựa trên SSL để cải thiện độ chính xác phân loại với số lượng mẫu được gắn nhãn hạn chế. Phương pháp được đề xuất sử dụng Laddernet làm bộ phân loại.
* Các đóng góp chính của bài báo:
* Đề xuất phương pháp ConvLaddernet dựa trên SSL: để giải quyết vấn đề độ chính xác thấp khi phân loại trong trường hợp ***ít mẫu có nhãn*** (ít hơn **20%** tổng số mẫu được dán nhãn).
* Đề xuất KT-ConvLaddernet method (knowledge transfer ConvLaddernnet): để giải quyết vấn đề độ chính xác thấp khi phân loại trong trường hợp ***ít mẫu có nhãn hơn*** (ít hơn **5%** tổng số mẫu được dán nhãn).
* Sử dụng phương pháp KTDA-CNN và KTDA-Convladdernet (DA ở đây là *domain adaptation*) để giải quyết vấn đề không phù hợp giữa miền nguồn và miền đích trong quy trình transfer learning.

### **Mô hình hệ thống**



Các máy tính và điện thoại di động khác nhau tạo ra lưu lượng truy cập mạng, được router thu thập và gửi đến trung tâm cơ sở dữ liệu. Trung tâm cơ sở gửi dữ liệu đến cloud server, nơi dữ liệu được xử lý. Phía bên trái của cloud server hiển thị tình huống cơ bản của tập dữ liệu, trong đó phần màu cam biểu thị dữ liệu được gắn nhãn và phần màu xanh biểu thị dữ liệu chưa được gắn nhãn. Dữ liệu được xử lý và chuyển đổi tương ứng thành định dạng hình ảnh và phương pháp SSL được sử dụng để thu được kết quả phân loại cuối cùng. Cụ thể, giai đoạn thu thập và xử lý traffic diễn ra như sau:

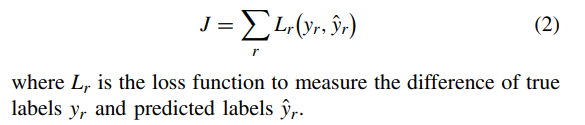
* Đầu tiên, tiền xử lý software traffic data: Đối với một nhóm mẫu lưu lượng truy cập phần mềm đã được gắn nhãn, phân loại CNN biểu diễn dưới dạng:



Trong đó:

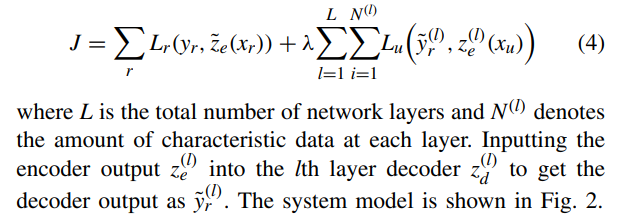
* z(·) đại diện cho hàm trích xuất feature.
* đại diện cho tham số mạng.
* yˆ : là kết quả dự đoán đã được phân loại của mạng.
* xr: dữ liệu được dán nhãn.
* yr: biểu thị các nhãn thực sự.

Trong backward propagation optimization (quá trình tối ưu hóa trọng số cho mô hình), objective function được biểu diễn:

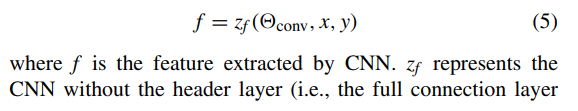


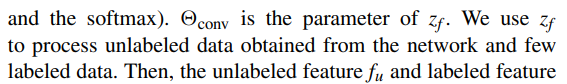
* Trong môi trường thực tế ít mẫu, phương pháp phân loại được CNN sử dụng không hiệu quả do thiếu dữ liệu huấn luyện. Do đó, bài báo sử dụng phương pháp SSL để tận dụng tối đa lượng lớn dữ liệu chưa được gắn nhãn tồn tại trong mạng. Trong SSL, mục đích của phần được giám sát của mạng và phần không được giám sát sẽ gắn nhãn không giống nhau, để khắc phục nó, bài báo thêm nhiễu Gaussian vào lớp mã hóa của bộ mã hóa và đặt phép biến đổi nhiễu là **g**, sau đó là Sử dụng **z˜e** và **ze** để xử lý dữ liệu được gắn nhãn và không được gắn nhãn.

Sau đó, để tối ưu hóa từng lớp của mạng, bài báo tính toán (là hàm mất mát của phần không được giám sát) cho mỗi lớp cần tối ưu hóa. Đối với bất kỳ lớp thứ **l (*l*th)** nào trong CNN, bài báo sử dụng hàm đo lường như sau:

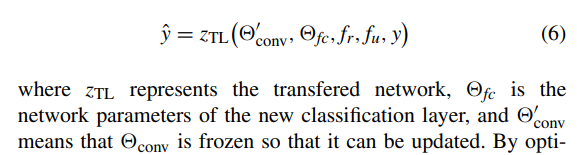


* Để tận dụng tối đa một số bộ dữ liệu hiện có, bài báo đề xuất transfer-SSL. Đầu tiên, bộ dữ liệu lưu lượng truy cập mạng hiện có được xử lý trước và sau đó được huấn luyện theo (2) để thu được CNN **z**, có thể được sử dụng để trích xuất các feature lưu lượng mạng. Sau đó, loại bỏ lớp phân loại của **z** và làm cho nó xuất ra dữ liệu đặc trưng như sau:



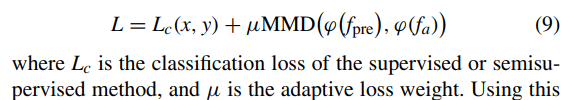


* Lấy và làm input, việc phân loại sẽ được biểu diễn như sau:



Cuối cùng, sử dụng (4) để tối ưu hóa và thu được mục tiêu phân loại.

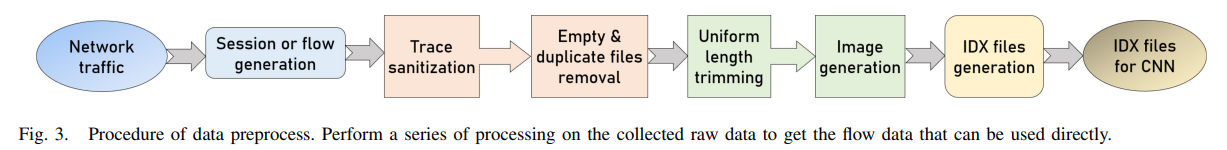
* Transfer learning đôi khi gặp phải vấn đề là việc phân bổ miền nguồn và miền đích khá khác nhau, để đạt được hiệu suất tốt hơn, tác giả sẽ sử dụng thích ứng miền DA. Các mẫu miền nguồn và mẫu miền đích được điều chỉnh theo cùng một không gian đặc trưng bằng ánh xạ ϕ, để nhận ra sự căn chỉnh. Bằng cách sử dụng hàm mất mát sau:



Quá trình huấn luyện có thể làm cho mô hình được huấn luyện kế thừa hiệu suất của mô hình được huấn luyện trước ở mức tối đa và đồng thời, thích ứng với trường hợp có ít mẫu được gắn nhãn.

## Phương pháp chính được sử dụng trong bài báo

### Tiền xử lý dữ liệu

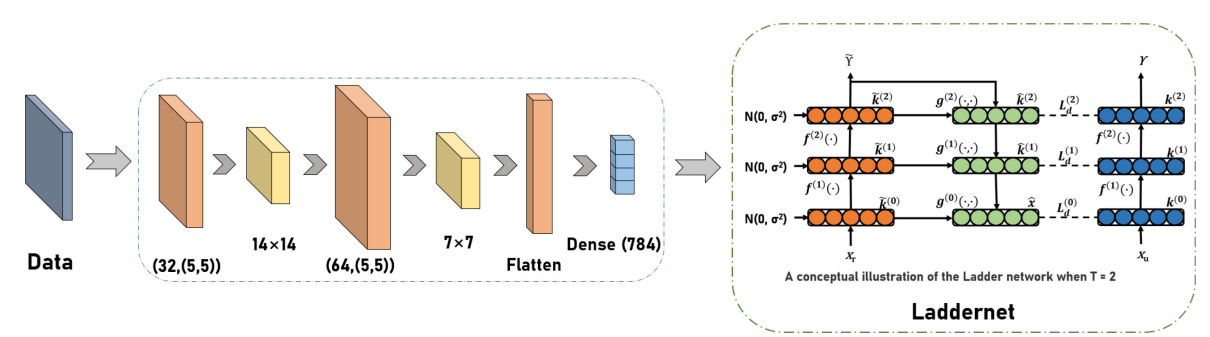


* Sơ đồ trên mô tả lại quá trình xử lý dữ liệu thô (raw data) để thu được định dạng dữ liệu phù hợp với yêu cầu của mô hình và có thể sử dụng trực tiếp để training cho CNN.
* Bộ dữ liệu được sử dụng trong thử nghiệm là USTC-TFC2016.
* Dataset gồm 2 phần: mười loại lưu lượng phần mềm độc hại được các nhà nghiên cứu thu thập từ môi trường mạng thực và phần còn lại là mười loại lưu lượng thông thường được thu thập bởi nền tảng mô phỏng lưu lượng mạng chuyên nghiệp IXIA BPS.
* Dữ liệu được cắt lát, đối chiếu để tránh trùng lặp dữ liệu (bằng công cụ fdupes), tạo ra hình ảnh với kích thước là 784 byte, sau đó được chuyển đổi thành định dạng IDX dùng cho CNN model (được sử dụng trong các mô hình cụ thể sẽ được trình bày ở phần sau).

### Quá trình đào tạo các model

#### **Model 1: ConvLaddernet**

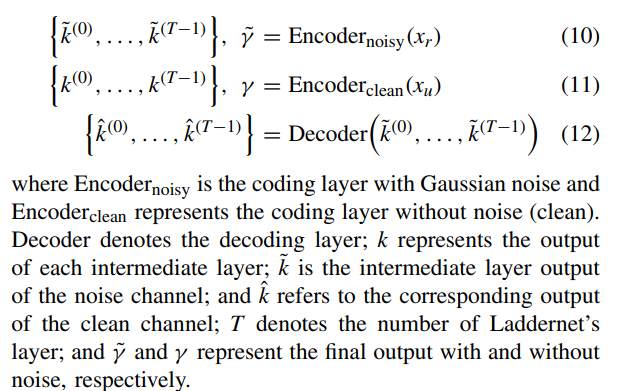
1. **Cấu trúc của ConvLaddernet**



Cấu trúc mạng nơ-ron bao gồm một convolutional layer (lớp tích chập) và một Laddernet. Kích thước dữ liệu đầu vào là 784 byte (28 × 28), số kênh trong lớp tích chập đầu tiên là 32, kích thước của convolution kernel là 5 × 5 và nó đi qua lớp tích chập đầu tiên. Sau đó kích thước dữ liệu trở thành 14 × 14, lớp chập thứ hai có 64 kênh, kích thước của convolution kernel vẫn là 5 × 5, ta thu được dữ liệu có kích thước 7 × 7. Tiếp tục, dữ liệu sẽ đi qua lớp flatten, ta thu được mảng một chiều làm đầu vào cho dense layer. Cuối cùng, dữ liệu sẽ được đưa vào Laddernet để tiến hành thêm nhiễu Gaussian vào ảnh để làm cho model học được cách biểu diễn ổn định hơn đối với ảnh nhiễu.

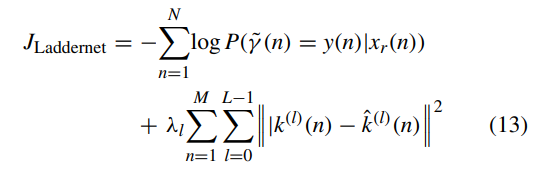
1. **Basic Principle of Laddernet**

* Điều đặc biệt nên sử dụng Ladder network vì nó phù hợp với transfer learning, do đó sẽ phù hợp với yêu cầu đặt ra trong paper này - đó là đào tạo mô hình trong tình huống thiếu dữ liệu có gắn nhãn. Cấu trúc của Ladder Network cung cấp khả năng chuyển giao kiến thức từ các tác vụ không giám sát sang các tác vụ giám sát mà không cần phải tinh chỉnh lại toàn bộ mô hình.
* Laddernet là một phương pháp SSL. Quan sát dạng toán học của Laddernet:



Là dữ liệu chưa dán nhãn, là dữ liệu đã dán nhãn.

* Laddernet sử dụng kênh nhiễu **γ˜** và nhãn thực **y** và đầu ra của mỗi lớp Encodeclean và Decoding, để cùng nhau xây dựng objective function:

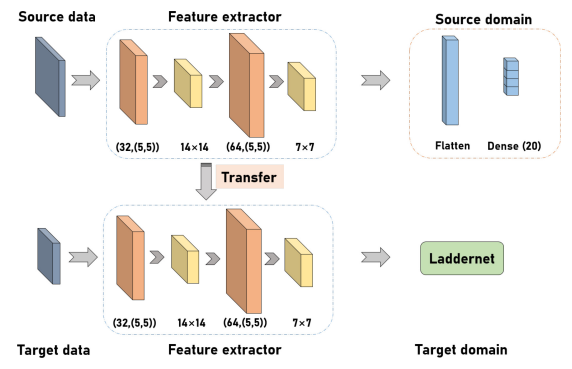


1. **Training and Test Process of ConvLaddernet**

* Tập dữ liệu được chia thành training và phần validation. Sau đó một phần mẫu nhỏ được tách ra thành dữ liệu được gắn nhãn trong phần đào tạo và phần còn lại của dữ liệu được sử dụng như dữ liệu không được gắn nhãn.
* Sử dụng trình tối ưu hóa Adam, learning rate là 0.02. Maximum epoch và batch size lần lượt là 50 and 128. Cuối cùng, sử dụng test set để đánh giá hiệu suất mô hình.

#### **Model 2: KT-ConvLaddernet**

1. **Structure of KT-ConvLadderne**



Sự khác biệt chính giữa ConvLaddernet và KT-ConvLaddernet là KT-ConvLaddernet sử dụng phương thức *transfer learning*. Đầu tiên, nhập dữ liệu miền nguồn vào pre - train network. Sau khi hoàn thành quá trình này, trình trích xuất tính năng sẽ chuyển đến miền đích và bị đóng băng. Sau đó, dữ liệu miền mục tiêu được đưa vào và trình trích xuất tính năng được kết nối với Laddernet để phân loại.

1. **Training and Testing Process of KT-ConvLaddernet**

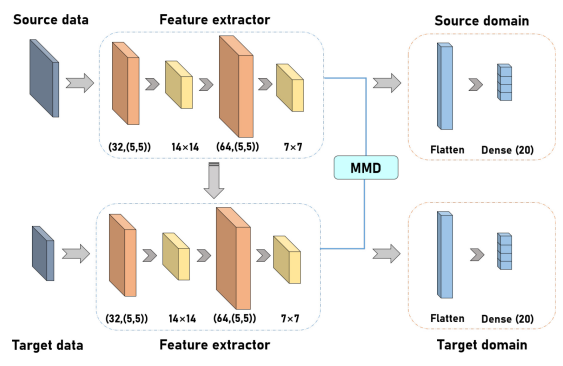
Trước tiên bài báo huấn luyện pretraining network sử dụng pretraining set. Sau đó chia dataset như mục III để hoàn thành huấn luyện chuyển giao. Maximum epoch trong pretraining và transfer training là 50, các thông số còn lại được đặt như mục III.

#### **Model 3: KTDA-CNN**

1. **Basic Principle of domain adaptation**

Ý tưởng của *domain adaptation* không phải là sử dụng mô hình miền nguồn để xử lý trực tiếp dữ liệu miền mục tiêu mà là căn chỉnh các feature mẫu của miền nguồn và miền mục tiêu càng nhiều càng tốt khi đào tạo mô hình miền mục tiêu, để đào tạo một mô hình miền mục tiêu mạnh mẽ.

1. **KTDA-CNN Method**

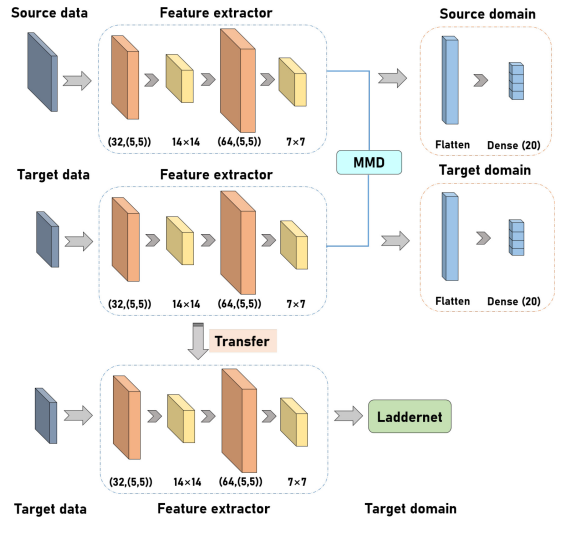
****

KTDA-CNN bao gồm hai lớp chập và hai lớp phân loại. Sự khác biệt chính giữa phương pháp này và KT-ConvLaddernet là nó không còn sử dụng phương thức truyền trực tiếp mà sử dụng phương pháp DA để truyền kiến thức cần thiết giữa lớp trích xuất tính năng được đào tạo trước và mạng mới.

MMD distance là một hình thức để hiện thực hóa DA. MMD là mất mát giữa lớp chập được huấn luyện trước và lớp chập huấn luyện mới, do đó MMD sẽ giúp các *feature* của lớp chập mới có thể gần với lớp chập được huấn luyện trước. Và dense layer được thêm vào sau *trained feature extraction layer*.

Tham số regularization hyperparameter **μ** được đặt là 0.25.

#### **Model 4: KTDA-ConvLaddernet**

****

KTDA-ConvLaddernet đã bổ sung thuật toán DA trên cơ sở KT-ConvLaddernet ban đầu. Do tính chất phức tạp của mô hình, bài báo sẽ triển khai gồm 3 giai đoạn đào tạo. Đầu tiên, *feature* của miền nguồn và miền mục tiêu được so khớp bằng cách thêm phép đo MMD để thu được lớp trích xuất có các *feature* gần giống nhau. Sau khi lớp trích xuất *feature* đã được huấn luyện được chuyển đến miền mục tiêu, một số mẫu được gắn nhãn sẽ được nhập vào để phân loại và kết quả phân loại sẽ thu được thông qua Ladder network.

#### **Các model khác**

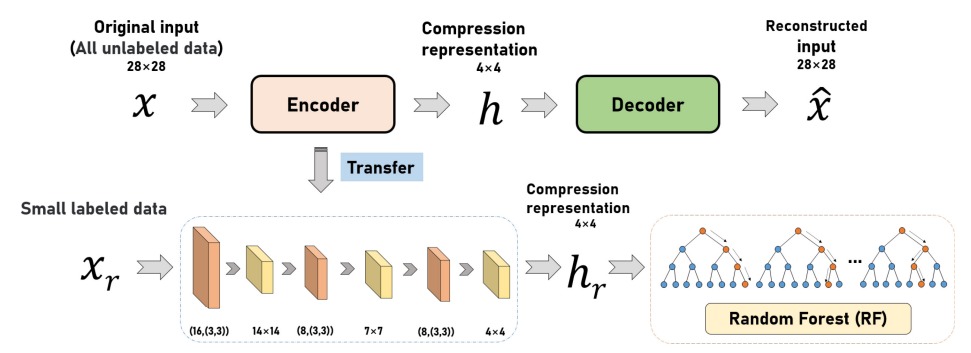
* Deep Layers MTC Methods:

Bài báo đã áp dụng cấu trúc mạng CNN tương đối đơn giản làm lớp trích xuất đặc trưng. Tuy nhiên, vì nhiều hạn chế nên không thể giải quyết chính xác vấn đề phân loại lưu lượng phần mềm độc hại. Vì vậy, mạng VGG sâu được đề xuất thay thế mạng CNN nông để đạt được độ chính xác phân loại cao hơn. Cụ thể sẽ được trình bày thông qua 2 phương pháp là KTDA-VGG và KTDA-VGGLaddernet.

* Benchmark Methods (phương pháp điểm chuẩn):
* CNN-Based Methods:

Tác giả huấn luyện một mạng CNN đơn giản để phân loại dữ liệu lưu lượng truy cập phần mềm. Nó bao gồm một mô-đun trích xuất feature với hai lớp tích chập. Các filter của nó lần lượt là 32 và 64. Dropout được sử dụng để ngăn chặn tình trạng quá khớp (overfitting) trong mỗi lớp chập.

* CAE With RF Classifier:

****

Trước tiên, huấn luyện một bộ mã hóa tự động (*autoencoder*) với tất cả dữ liệu lưu lượng truy cập phần mềm không được gán nhãn. Nó chứa tất cả feature của dữ liệu thô.

Sau đó, gửi mẫu dữ liệu nhỏ vào trong *trained autoencoder* để thu được feature **h** output của bộ mã hóa. Feature **h** sẽ được sử dụng như là nhãn để huấn luyện bộ phân loại. Lưu lượng phần mềm được mã hóa bởi bộ mã hóa tự động sẽ có những *key features*, do đó sẽ cải thiện được hiệu suất cho RF trong trường hợp ít mẫu có nhãn.

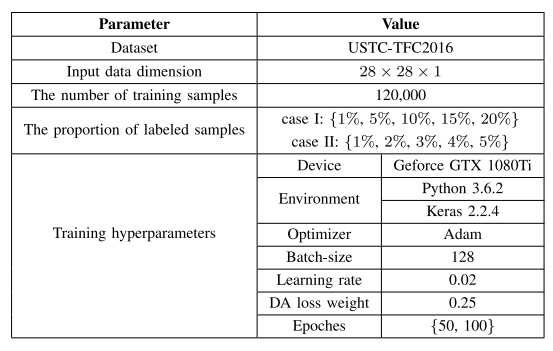
* Joint Transferring and Classification Method:

Trong giai đoạn thứ hai của phương pháp này, DA được sử dụng để huấn luyện mạng trích xuất đặc trưng và Laddernet cho phân loại sẽ được đào tạo với vài mẫu.

## Kết quả thực nghiệm của bài báo

### Môi trường thực nghiệm

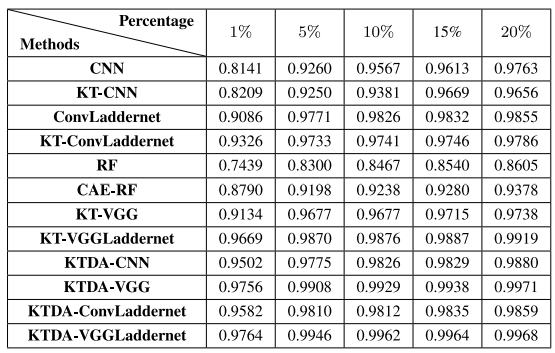
Máy chủ của bài báo sử dụng GPU Geforce GTX 1080ti với 11 RAM cho máy tính. Môi trường là Keras 2.2.4 deeplearning framework và Python 3.6.2. Sử dụng các công cụ trong sklearn 0.23.2 để đánh giá hiệu suất mô hình. RF model được sử dụng trong thí nghiệm cũng được gọi là sklearn. Chi tiết được thể hiện trong bảng:

****

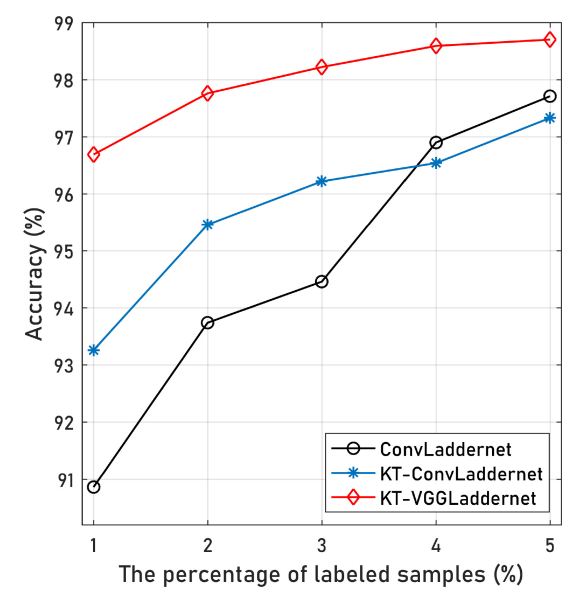
### Kết quả thực nghiệm

1. **Experimental Results Based on ConvLaddernet and KT-ConvLaddernet**

* Có 2 so sánh sẽ được tiến hành:
* ConvLaddernet, KT-ConvLaddernet với CNN, KT-CNN, và RF trong trường hợp {1%, 5%, 10%, 15%, 20%} các mẫu dán nhãn.
* Sau đó, so sánh với ConvLaddernet và KT-ConvLaddernet trong trường hợp {1%, 2%, 3%, 4%, 5%}.



* Theo kết quả ở bảng trên cho thấy, phương pháp với *knowledge transfer learning* (KT-CNN và KT-ConvLaddernet) đạt được hiệu suất tốt hơn trong trường hợp có ít mẫu hơn (1%–5%). Các phương pháp trực tiếp trích xuất đặc trưng (CNN và ConvLaddernet) đạt kết quả tốt hơn khi số lượng mẫu được dán nhãn tăng dần (5%-20%).
* Thí nghiệm bổ sung cho trường hợp tỉ lệ mẫu có dán nhãn là 1%-5%:
* Dưới 3.8%: hiệu suất của KT-ConvLaddernet tốt hơn ConvLaddernet.
* Trên 3.8%: Convladdernet cho kết quả tốt hơn.



Kết luận, *transfer learning* phù hợp hơn với số lượng mẫu ít hơn.

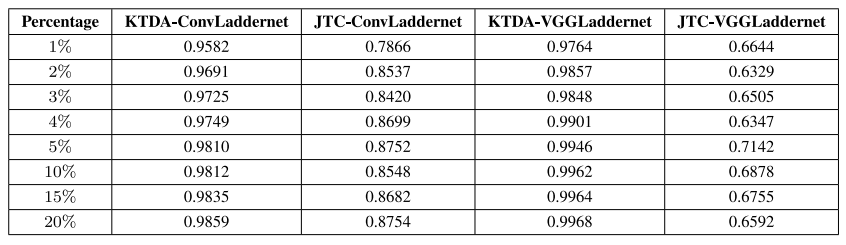
* So sánh phương pháp bài báo đề xuất (ConvLaddernet) với các phương pháp chuẩn (CNN, CAE-RF):
* CAE-RF có độ chính xác phân loại cao hơn RF, nhưng độ chính xác phân loại của nó không tốt bằng sử dụng trực tiếp mạng CNN.
* Trong trường hợp tỷ lệ mẫu là 1% thì độ chính xác phân loại của việc sử dụng Laddernet cao hơn 9,45% so với CNN ban đầu. Do đó, đã chứng minh ConvLaddernet hiệu quả hơn trong trường hợp có ít mẫu hơn.

1. **Experimental Results Based on KTDA Methods**

* Tiến hành so sánh:
* So sánh hiệu suất của KTDA với các phương pháp ConvLaddernet và KTConvLaddernet trong trường hợp có ít mẫu (5%–20%) và ít mẫu hơn (1%–5%). Tiến hành thử nghiệm cho deep VGG network.
* So sánh hiệu suất của KT DA-ConvLaddernet với phương pháp so sánh JTC.
* Kết quả so sánh:
* KTDA-CNN có độ chính xác phân loại cao hơn KT-CNN trong cả 2 trường hợp (bảng kết quả tại mục III.2a).
* KTDA-ConvLaddernet trong 2 trường hợp cao hơn so với phương pháp KT-ConvLaddernet và ConvLaddernet. Cùng tỉ lệ 15%, KTDA-ConvLaddernet cao hơn 0.89% so với KT-ConvLaddernet. Cùng tỉ lệ 1%, KTDA-ConvLaddernet cao hơn 2.56% so với KT-ConvLaddernet.

Tóm lại, KTDA đã cải thiện đáng kể hiệu suất so với KT-ConvLaddernet và ConvLaddernet, nhất là khi tỉ lệ mẫu dán nhãn nhỏ hơn.

* Độ chính xác phân loại của deep VGG network: Với tỷ lệ phần trăm mẫu gán nhãn ít hơn, sự cải thiện hiệu suất này rõ ràng hơn.
* Cuối cùng, so sánh hiệu suất của KTDA-ConvLaddernet và JTC-ConvLaddernet:



Có thể thấy, độ chính xác phân loại của phương pháp JTC ConvLaddernet thấp hơn nhiều so với phương pháp KTDA-ConvLaddernet.

Ngoài ra, hiệu suất của JTC-ConvLaddernet khi sử dụng mạng sâu thấp hơn khi sử dụng mạng nông. Điều này là do độ phức tạp của mạng hai giai đoạn quá cao, khiến cho đào tạo không thể hội tụ một cách hiệu quả.

### Kết luận của bài báo

* Trong bài viết này, đề xuất hai phương pháp *malware traffic classification* (MTC) dựa trên ConvLaddernet và KT-ConvLaddernet tương ứng cho các ứng dụng của IIoT. Phương pháp MTC dựa trên ConvLaddernet đạt được hiệu suất tốt hơn với số lượng ít mẫu được gắn nhãn (20%). Trong khi KT-ConvLaddernet đạt hiệu suất tốt hơn với ít mẫu được gán nhãn hơn (5%) mẫu được gắn nhãn.
* Phương pháp CAE-RF được đề xuất như một phương pháp so sánh. Kết quả thực nghiệm cho thấy trong trường hợp ít mẫu được gắn nhãn hơn, việc trích xuất các đặc điểm mẫu trước khi huấn luyện có thể đạt được độ chính xác phân loại cao hơn.
* Kết quả thực nghiệm cho thấy độ chính xác của phương pháp KTDA-ConvLaddernet tốt hơn ConvLaddernet và phương pháp KT-ConvLaddernet ở cả 2 trường hợp. Và sau khi thay thế mạng CNN nông bằng mạng deeper VGG, hiệu suất phân loại của mô hình đề xuất có thể được cải thiện hơn nữa cho secure IIoT.
* Các mô hình được sử dụng trong bài viết này là các mô hình phổ biến trong lĩnh vực nhận dạng hình ảnh và có độ phức tạp cao. Do đó, khi thực hiện một số nhiệm vụ cụ thể, cần phải đo lường độ chính xác và khả năng tính toán và mô hình phải được đơn giản hóa một cách thích hợp.
* Các phương pháp dựa trên transfer learning được đề xuất đều yêu cầu các mô hình tiền huấn luyện, đôi khi khó triển khai trong các ứng dụng thực tế.

## Kết quả thực nghiệm của nhóm

Do paper đề cập đến khá nhiều model, nhưng trong phạm vi môn học, môi trường thực nghiệm hạn chế và thời gian không cho phép, nên nhóm em sẽ chỉ training trên 4 model, bao gồm: KT-CNN, KT-ConvLaddernet, KTDA-CNN, KTDA-ConvLaddernet vì đây là 4 model chính và được tác giả khẳng định là đem lại hiệu suất tốt hơn so với các mô hình khác trong điều kiện thiếu dữ liệu gán nhãn.

### Môi trường thực nghiệm

* Sử dụng laptop cá nhân để thực hiện đồ án.
* Giai đoạn xử lý dataset được tiến hành trên máy ảo Ubuntu 22.04.3. Tuy nhiên do lượng dữ liệu lớn nên mất nhiều thời gian để xử lý. Do đó nhóm em đã chuyển sang thực nghiệm trên máy thật. Vì vậy giai đoạn training mô hình được tiến hành trên Windows 11 bằng công cụ VSCode.

### Xử lý dataset

Thực hiện trên máy ảo Ubuntu 22.04.3:

1. **Download các phụ thuộc:**

* Download Mono: Mono là một nền tảng phát triển và một bộ công cụ cho việc phát triển và triển khai ứng dụng dựa trên ngôn ngữ lập trình C# . Mono giúp mở rộng khả năng chia sẻ mã nguồn giữa các hệ điều hành và nền tảng khác nhau, tạo điều kiện thuận lợi cho việc phát triển ứng dụng đa nền tảng trong môi trường .NET.

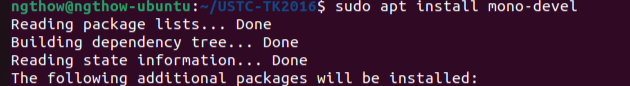
Trước hết, cần cài đặt *libpng12-0 deb* (là một gói thư viện (library package) trong Linux, chủ yếu được sử dụng để hỗ trợ xử lý ảnh PNG):

*sudo add-apt-repository ppa:linuxuprising/libpng12*

*sudo apt update*

*sudo apt install libpng12-0*

Sau đó, cài đặt mono:



* Cài đặt powershell core: là một phiên bản mở rộng và đa nền tảng của PowerShell. PowerShell Core được thiết kế để chạy trên nhiều hệ điều hành và nền tảng khác nhau, bao gồm Windows, Linux, và macOS.
* Thêm Microsoft Package Repository:

*wget -q https://packages.microsoft.com/config/ubuntu/$(lsb\_release -rs)/packages-microsoft-prod.deb -O packages-microsoft-prod.deb*

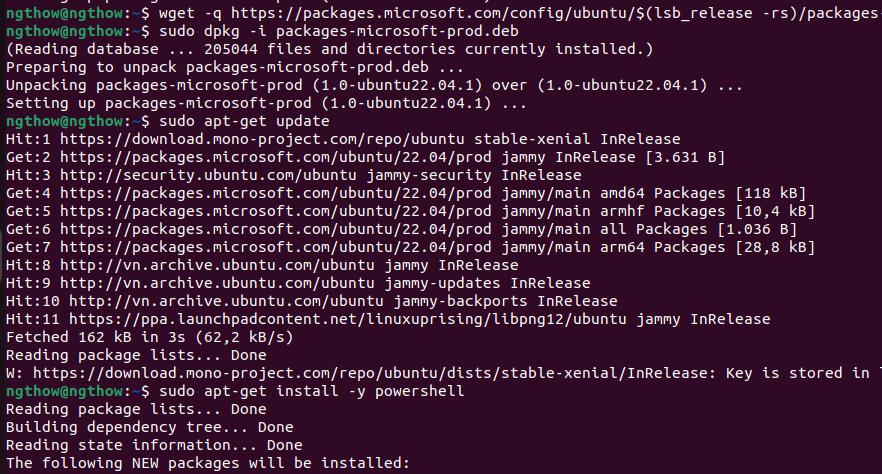
*sudo dpkg -i packages-microsoft-prod.deb*

* Cập nhật Package List:

*sudo apt-get update*

* Cài đặt PowerShell Core:

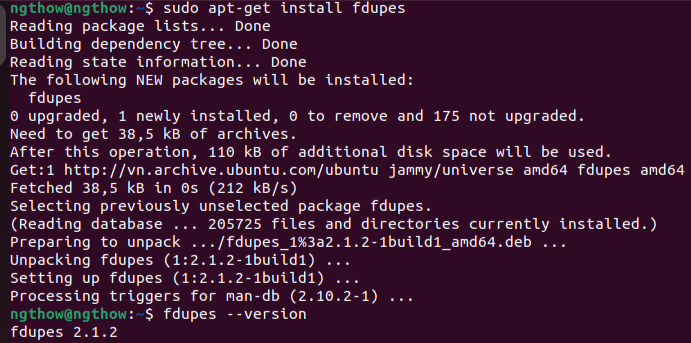
*sudo apt-get install -y powershell*



* Để chạy PowerShell Core, sử dụng lệnh pwsh:



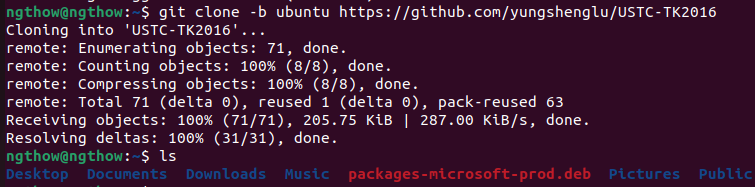
* Cài đặt fdupes (nó là một trong những công cụ đơn giản nhất để định vị và loại bỏ các tệp trùng lặp trong thư mục):



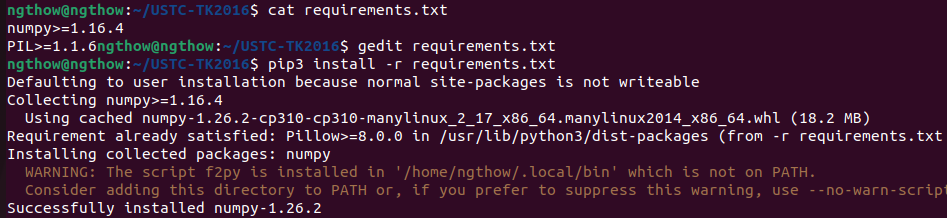
1. **Download toolkit:**

* Tải xuống toolkit có tên là "USTC-TK2016", được sử dụng để phân tích lưu lượng mạng (tệp .pcap). Link toolkit:

<https://github.com/yungshenglu/USTC-TK2016>



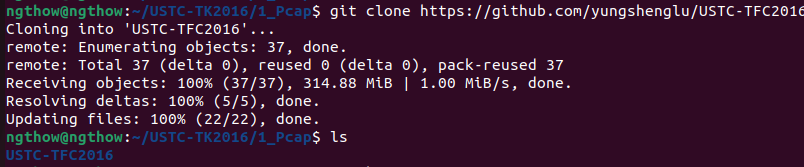
* Cài đặt các gói cần thiết:



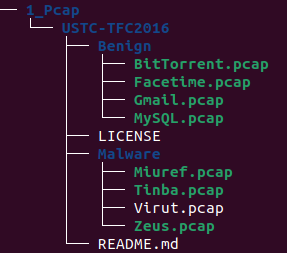
Do PIL hiện tại không còn được hỗ trợ nên em sẽ cài đặt gói thay thế khác là *Pillow>=8.0.0.*

1. **Execution:**

* Download traffic dataset USTC-TFC2016 tại <https://github.com/echowei/DeepTraffic/tree/master/1.malware_traffic_classification/1.DataSet(USTC-TFC2016)> và đặt nó vào thư mục 1\_Pcap/:

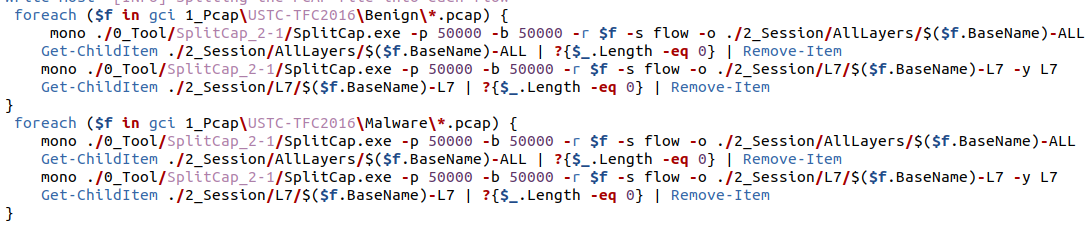


Cấu trúc thư mục 1\_Pcap/ như hình (do máy ảo chạy lâu nên em đã loại bỏ một số loại malware và benign, tuy nhiên do chuyển sang train model trên máy thật – có tốc độ xử lí nhanh hơn, nên em đã sử dụng tất cả các loại traffic software có trong dataset, và kết quả thực nghiệm của nhóm là kết quả được đánh giá trên toàn bộ tập dataset USTC-TFC2016):



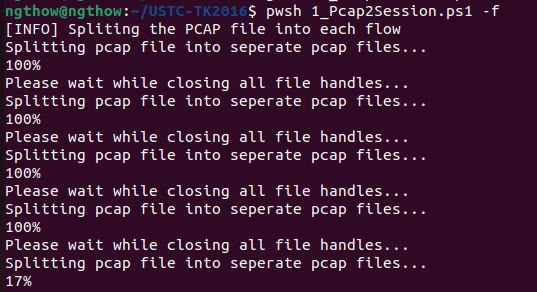
* Chia dataset theo session và flow:
* Cần chia theo session để model có thể nắm được thông tin toàn diện về hành vi của thiết bị mạng, giúp theo dõi hành vi của người dùng từ lúc bắt đầu đến khi kết thúc một session. Chia theo flow có thể bao gồm các thông tin về port, giao thức, thời gian truyền tải và các thông tin khác liên quan đến giao tiếp mạng. Chia theo session và flow nhằm mục đích theo dõi, phát hiện hành vi bất thường, hoặc giúp dự đoán các hành vi trong tương lai.
* Mở terminal và chạy file 1\_Pcap2Session.ps1:

Do file trên do tác giả cung cấp có sai đường dẫn và version không phù hợp, nên cần chỉnh sửa lại như sau. Ta phải sửa nội dung file để trỏ trực tiếp đến thư mục chứa các file PCAP:



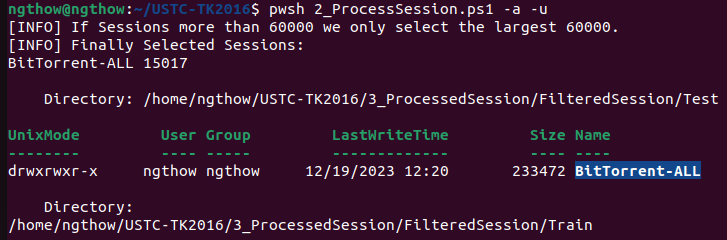
Và tải phiên bản mới nhất của SplitCap để không xảy ra lỗi (tải tại link sau <https://www.netresec.com/?page=SplitCap> và sau đó thay thế file SplitCap.exe trong thư mục 0\_Tools thành file .exe vừa tải về).

* Chia các file .pcap theo flow bằng câu lệnh như trong hình (theo session tương tự, nhưng đổi option thành -s):



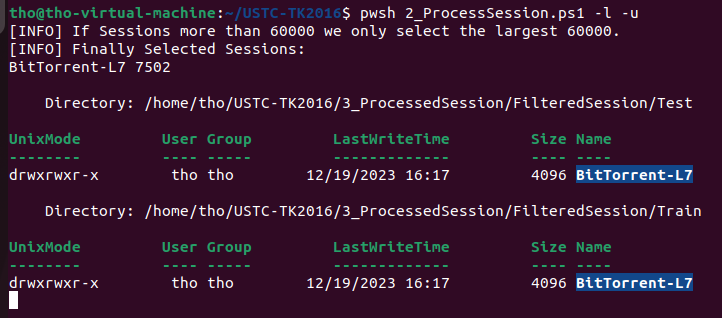
* Thực thi file 2\_ProcessSession.ps1:
* Xử lý tệp PCAP với tất cả các lớp (ALL) (unsort):

*pwsh 2\_ProcessSession.ps1 -a -u*



* Chỉ xử lý tệp PCAP với lớp 7 (L7) (unsort):

*pwsh 2\_ProcessSession.ps1 -l –u*

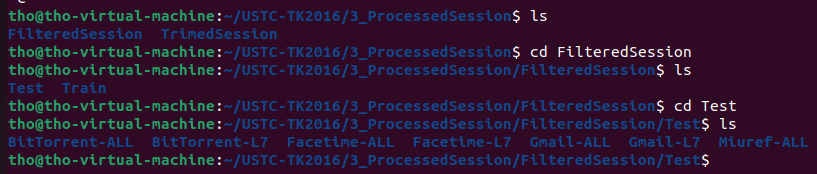


* Nếu thành công, ta sẽ thấy 2 thư mục sau trong thư mục 3\_ProcessedSession/:

FilteredSession/ - Get the top 60000 large PCAP files

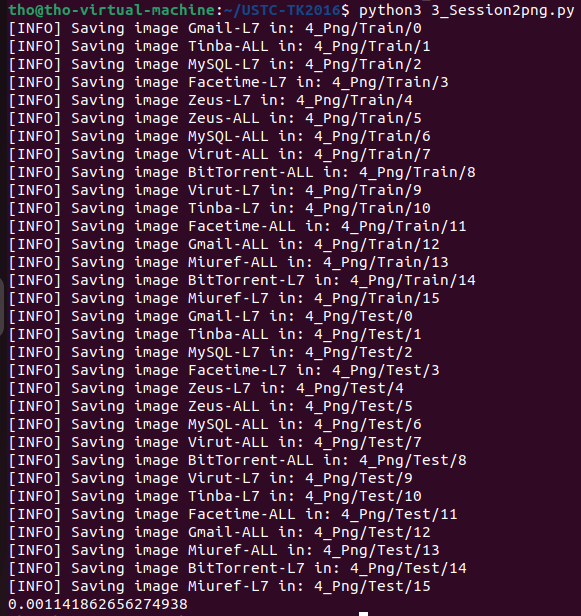
TrimedSession/ - Trim the filtered PCAP files into size 784 bytes (28 x 28) and append 0x00 if the PCAP file is shorter than 784 bytes.

Các file trong thư mục con Test/ và Train/ được lấy ngẫu nhiên từ dataset.



* Thực thi file 3\_Session2Png.py:

*python3 3\_Session2png.py*

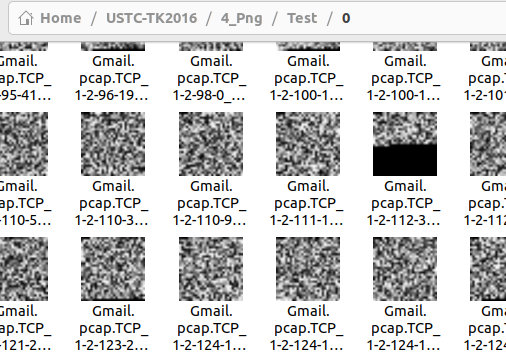


Nếu thành công, ta sẽ thấy 2 thư mục sau trong thư mục 4\_Png/

Test/ - For testing

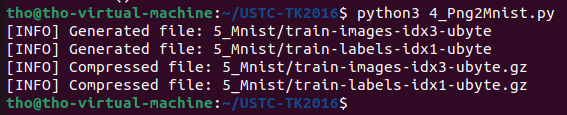
Train/ - For training

Dữ liệu ở dạng hình ảnh, định dạng tương tự như sau:



* Thực thi file 4\_Png2Mnist.py:

*python3 4\_Png2Mnist.py*



Khi thành công, ta sẽ thấy các file/thư mục sau trong thư mục 5\_Mnist/

train-images-idx1-ubyte

train-images-idx3-ubyte

train-images-idx1-ubyte.gz

train-images-idx3-ubyte.gz

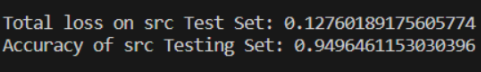
### Kết quả thực nghiệm

* Link github chứa model:<https://github.com/yzjh/Keras-MTC-DA-Ladder>
* Giai đoạn thực nghiệm sẽ được tiến hành trên Windows 11 bằng công cụ VSCode.
* Nhóm em tiến hành thực nghiệm trên 4 model: KT-CNN, KT-ConvLaddernet, KTDA-CNN, KTDA-ConvLaddernet, với tỉ lệ mẫu có dán nhãn là 20%:



* Kết quả:
* 1. So sánh phương pháp với knowledge transfer learning (KT-ConvLaddernet) và phương pháp trực tiếp trích xuất đặc trưng (KT-CNN): KT-ConvLaddernet đạt hiệu suất tốt hơn khi train với tỉ lệ mẫu có nhãn là 20%:

KT-ConvLaddernet:

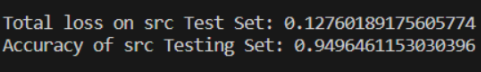


KT-CNN:

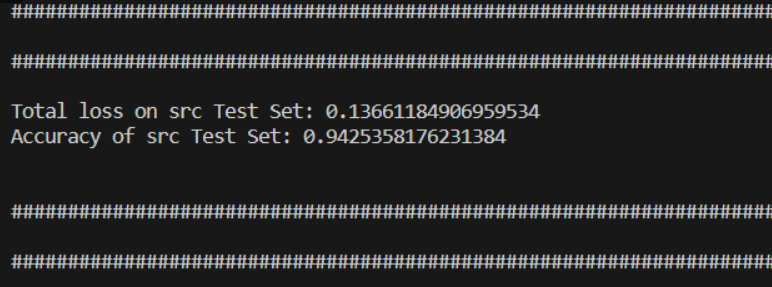


* 2. Dựa trên so sánh model có sử dụng thích ứng miền với model không sử dụng: Model có DA có accuracy tốt hơn, tuy nhiên mức độ loss cao hơn:

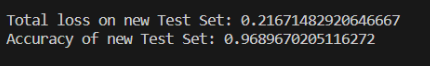
KT-ConvLaddernet:



KTDA-CNN:



KTDA-ConvLadderNet:



## Kết luận:

Thông qua quá trình thực nghiệm, nhóm đã nhận thấy rằng trong trường hợp số lượng mẫu gắn nhãn hạn chế, *domain adaptation* sẽ giúp cải thiện hiệu suất cho model. Đồng thời, phương pháp *knowledge transfer learning* cũng giúp model dự đoán chính xác hơn so với trích xuất feature trực tiếp.

Tuy nhiên, trong quá trình thực hiện đồ án, do chưa học qua kiến thức machine learning một cách có hệ thống nên dẫn đến việc kết quả thực nghiệm còn nhiều thiếu sót. Nhóm em mong nhận được lời nhận xét từ thầy Phạm Văn Hậu để chúng em có thể thực hiện tốt hơn trong những đồ án tiếp theo. Chúng em xin gửi lời cảm ơn đến thầy đã có những góp ý trong buổi báo cáo đồ án cuối kỳ vừa qua, cảm ơn thầy Hậu và thầy Nghi Hoàng Khoa đã hỗ trợ nhóm em trong quá trình thực hiện đồ án. Xin chân thành cảm ơn các thầy ạ!