BÁO CÁO ĐỒ ÁN MÔN HỌC

Môn học: Cơ chế hoạt động của mã độc

Tên chủ đề: Báo cáo đồ án môn học

GVHD: Phan Thế Duy

1. THÔNG TIN CHUNG:

Nhóm G23

Lớp: NT230.O21.ANTT

STT	Họ và tên	MSSV	Email
1	Nguyễn Thành Công	20521143	20521143@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá
1	Chuẩn bị dataset	100%
2	Chọn đặc tính	100%
3	Tiền xử lý các đặt tính	100%
4	Xây dựng mô hình học sâu	100%
5	Đánh giá mô hình học sâu	100%

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

Mục lục

1. Giới thiệu về đồ án

Tên đồ án tiếng anh: HYDRA: A multimodal deep learning framework for malware classification
Tiếng việt: HYDRA áp dụng nhiều mô hình máy học để phân loại mã độc.

Tóm tắt đồ án:

Trong việc phân loại các mã độc trong các file mã độc thực thi (ở dạng PE) thành các nhóm mã độc giúp tăng hiệu quả quá trình phân tích các file mã độc. Hơn nữa việc phân loại giúp hiểu hơn cách thức lây nhiễm các máy tính và các thiết bị từ đó đưa ra những chiến lược phù hợp để phát hiện và ngăn chặn các mã độc.

Với số lượng các tệp tin mã độc nhiều, việc phân loại một cách thủ công mất rất nhiều nguồn lực bao gồm con người và thời gian. Hơn nữa các biến thể của mã độc luôn biến đổi không ngừng khiến cho việc phân tích thủ công gặp nhiều thách thức. Việc ứng dụng học máy, học sâu sẽ giúp tự động hóa quá trình phân loại, đồng thời giúp giảm chi phí về con người, thời gian.

Thông qua bài nghiên cứu này, nhóm em đã thực hiện ứng dụng các mô hình học sâu dựa trên tập dữ liệu Microsoft Malware classfication. Hơn nữa, nhóm em cũng đã thực hiện thu thập các tệp tin mã độc dưới dạng file thực thi đã được gán nhãn, sau đó ứng dụng học sâu vào việc phân loại các tệp tin này.

Kết quả đạt được của nhóm em trên tập dữ liệu Microsoft Malware classfication nhóm em thực nghiệm trên 4 mô hình học sâu với các đặc trưng khác nhau và đạt kết quả Accuracy là 0.93 trên mô hình kết hợp (HYDRA). Trên dữ liệu thu thập tự thu thập nhóm em cũng đạt kết quả tương đối tốt.

2. Tập dữ liệu

Microsoft Malware classfication .
Kích thước khi giải nén là 1 nửa Terabyte.

Tập dữ liệu này chứa tổng 9 loại mã độc. Mỗi file mã độc được gán một mã hash duy nhất

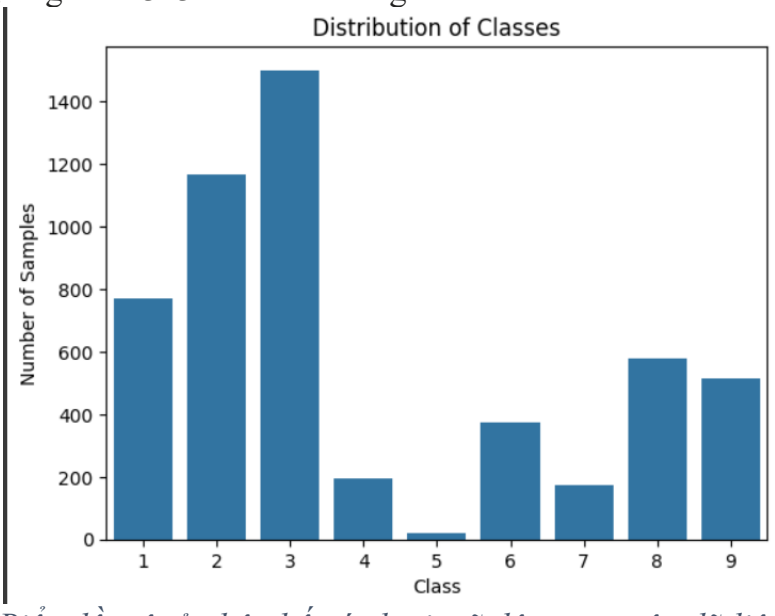
Family Name	# Train Samples	Type
Ramnit	1541	Worm
Lollipop	2478	Adware
Kelihos_ver3	2942	Backdoor
Vundo	475	Trojan
Simda	42	Backdoor
Tracur	751	TrojanDownloader
Kelihos_ver1	398	Backdoor
Obfuscator.ACY	1228	Any kind of obfuscated malware
Gatak	1013	Backdoor

Hình 1 Tổng quan về tập dữ liệu Microsoft Malware classification

Trong nghiên cứu thực nghiệm này, em chỉ sử dụng như sau:

Tổng tập dữ liệu là 6612 mẫu file asm và 6612 mẫu file bytes. Cả hai loại file này tương ứng với tên file là mã hash. Trong đó file asm được tạo ra bởi IDA Pro khi phân tích file mã độc. Tổng kích thước tập dữ liệu khi giải nén là gần 200 GB.

Tập dữ liệu trên được chia theo tỉ lệ là 80 % tương ứng với 5289 mẫu để huấn luyện và 20 % tương ứng với 1323 mẫu để đánh giá mô hình sau khi huấn luyện.



Hình 2 Biểu đồ mô tả phân bố các loại mã độc trong tập dữ liệu.

Như trong hình 2, thì tập dữ liệu có sự phân bố không đều, mất cân bằng, cho nên khi huấn luyện mô hình các loại mã độc có số mẫu ít, sẽ có sự chính xác không cao so với các loại khác.

a) Tập dữ liệu mã bytecode.

- Cách xử lý byte code là với mỗi file bytecode sẽ lấy trình tự các bytecode của mỗi file và lấy tối đa một ngưỡng là 2 triệu kí tự. Nếu các byte có kích thước nhỏ hơn thì sẽ được padding vào (chèn các kí tự 00 vào cuối).
- Đầu vào của mô hình huấn luyện byte code sẽ là các byte code theo trình tự.
- Sau đó, trước khi đưa vào đầu vào của lớp đầu vào, thì các byte code này sẽ được chuyển đổi thành một số tương ứng. Ví dụ, 0A là 0A, 0B là 11 v.v

b) Tập dữ liệu mã các mã lệnh hợp ngữ(mnemonic):

- Cách xử lý là sử dụng file asm rồi dùng kĩ thuật regular expresion để parse lấy các lệnh hợp ngữ như mov, inc, add v.v
- Với mỗi file này em sẽ lấy tuần tự một chuỗi các lệnh hợp ngữ với mỗi file asm sẽ lấy tối đa 50000 lệnh, tổng lệnh triết xuất ra ít hơn 50000 thì sẽ được padding (chèn lệnh NOP vào cuối file).
- Trước khi đưa vào lớp đầu tiên thì các mã lệnh hợp ngữ này sẽ được ánh xạ với các số tương ứng. Ví dụ mov sẽ là số 1, add là số 2 v.v

c) Tập dữ liệu API

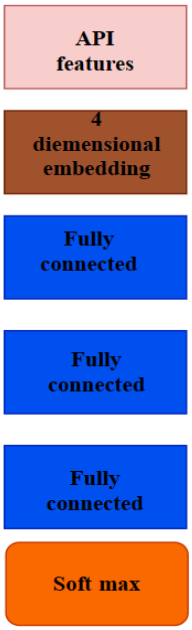
- Em sẽ sử dụng regular expresion để parse file asm để lấy ra các lời gọi API.
- Các API này khi đưa vào huấn luyện sẽ ở dạng vector với tương ứng là 0 và 1. Sẽ là 0 nếu lời gọi API được sử dụng và 1 nếu được sử dụng.

d) Tập dữ liệu cho model kết hợp (HYDRA)

Em sẽ sử dụng phương pháp như ba tập trên nhưng dữ liệu sẽ được kết hợp cả 3 mẫu trên.

3. Các thuộc tính dùng để huấn luyện dữ liệu

3.1 Hàm gọi Window API



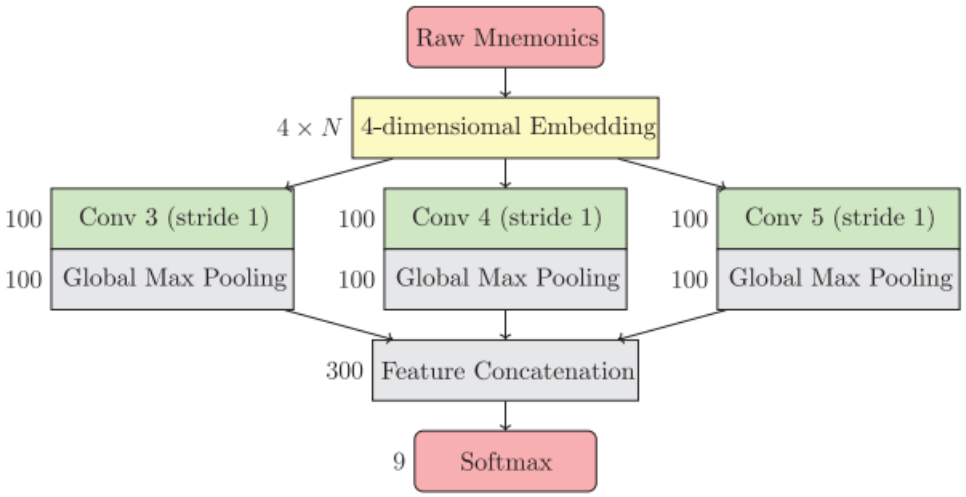
Hình 3 Mô hình lời gợi window api

Đầu vào là một vector K gồm các apiAPI call có các giá trị tương ứng là 0 hoặc 1. Giá trị là 1 nếu apiAPI call được sử dụng, 0 là không sử dụng.

Hàm activation softmax ở lớp cuối sẽ đóng vai trò phân loại các nhãn.

Activation của các lớp ẩn là ELU (Exponential Linear Unit).

3.2 Mnemonics analysis(phân tích các chuỗi lệnh hợp ngữ).



Hình 4 Kiến trúc mô hình lệnh hợp ngữ

Sử dụng CNN (convolutional neural network), một giải pháp hiệu quả thay thế cho phương pháp n-gram.

Mục đích là loại bỏ các quy trình truyền thống trong chiết xuất thuộc tính, lựa chọn thuộc tính, v.v.

Ưu điểm là không cần sử dụng các quy trình truyền thống, tiền xử lý dữ liệu. Chỉ cần đầu vào là chuỗi tuần tự các lệnh hợp ngữ.

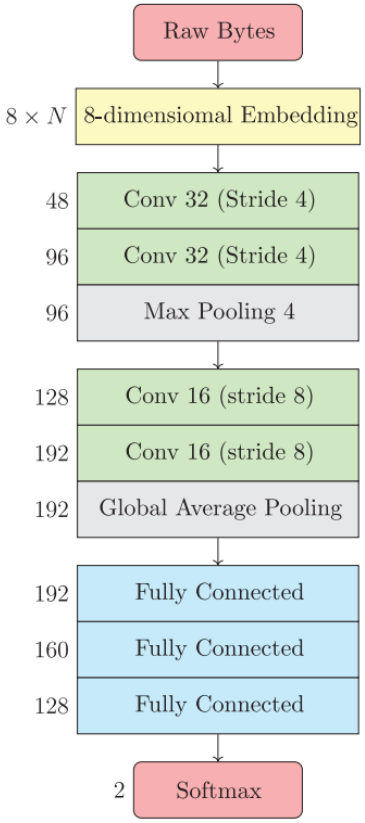
Nhược điểm: thời gian huấn luyện lâu.

Các lớp của mô hình:

- Lớp đầu vào:
  - Chuỗi tuần tự các lệnh hợp ngữ.
  - Sử dụng kĩ thuật one-hot encoding ánh xạ mỗi lệnh hợp ngữ với một số tự nhiên từ 1 tới I trong đó I là kích thước các lệnh hợp ngữ được sử dụng làm mẫu.
- Lớp nhúng:
  - Vì one-code encoding không phân tích được ngữ nghĩa của các chuỗi lệnh hợp ngữ. Nên ở lớp nhúng sử dụng kĩ thuật word-embedding
- Lớp tích chập:
  - Áp dụng nhiều bộ lọc các chuỗi hợp ngữ và chiết xuất n-gram. Kích thước bộ lọc là h *k trong đó h ∈ (Gibert, Mateu et al. 2020) và k là kích thước của lớp nhúng.
  - Hàm kích hoạt là ELU
- Lớp Global max-pooling layer: dùng để giảm số chiều vector nhưng vẫn giữ các thông tin quan trọng từ hàm kích hoạt.

- Lớp softmax: sử dụng thông tin các lớp trước để phân loại kết quả dự trên tỉ lệ.

3.3 Bytes analysis

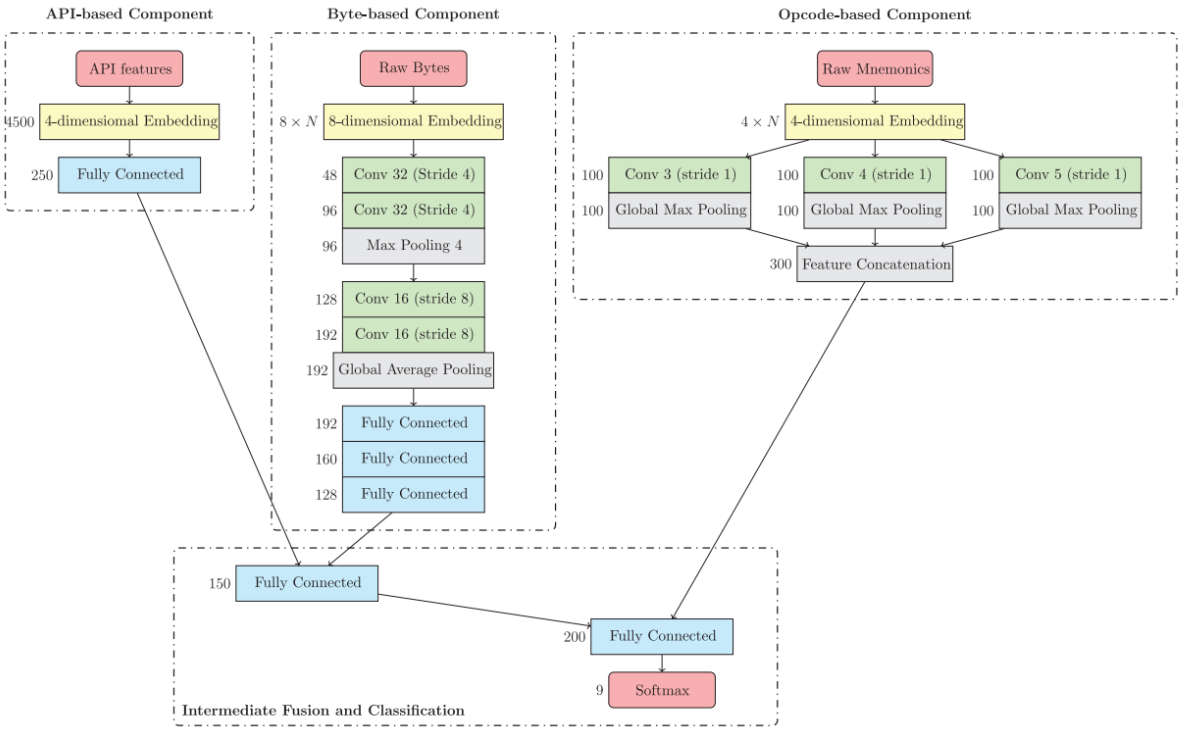


Hình 5 Kiến trúc mô hình Bytes

Lớp đầu vào là một chuỗi tuần tự các byte code. Với kích thước là 2 triệu byte cho một mẫu dữ liệu. Mỗi bytecode trong các mẫu này sẽ được ánh xạ và chuyển về số tự nhiên.

4. Huấn luyện đa mô hình HYDRA

4.1 Kiến trúc



Hình 6 Kiến trúc mô hình tổng thể

Mô hình tích hợp gồm 3 mô hình con kết hợp lại, bao gồm:

Mô hình API, mô hình Bytecode và mô hình Opcode. Các mô hình con này được chạy song song với nhau và chỉ được tích hợp lại ở 2 lớp cuối.

Việc kết hợp các mô hình con này sẽ làm tăng độc hiệu quả và chính xác và lấy được ưu điểm của mỗi mô hình.

4.2 Đưa tham số các mô hình học tối ưu ở mỗi mô hình vào mô hình tổng thể.

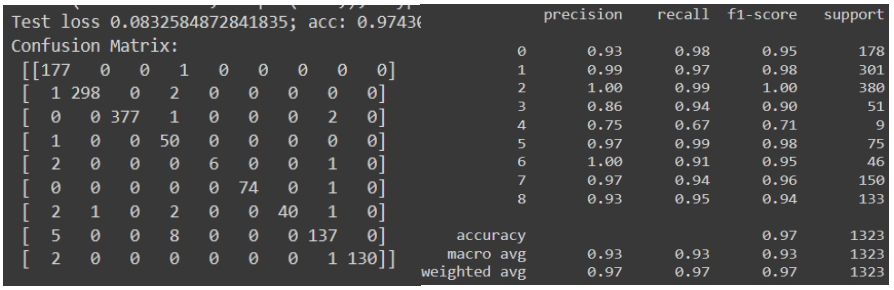
Để giảm thiểu thời gian huấn luyện trên mô hình HYDRA, em thực hiện đưa các tham số học đã được tối ưu trên các mô hình con ở các nhánh đơn đã được tối ưu làm tham số đầu vào cho mô hình HYDRA.

5. Phương pháp đánh giá kết quả và kết quả.

Phương pháp đánh giá là sử dụng điểm accuracy, f1-score, precision và recall

a) Accuracy: Kết quả này đánh giá độ phân loại chính xác của mô hình học sâu, có công thức là tổng giá trị thực sự là dương tính và âm tính được đánh giá đúng bởi mô hình chia cho tổng giá trị mẫu trong mô hình.
b) Precicion: công thức là tổng giá trị dương tính đúng chia cho tổng số được đánh giá là dương tính bởi mô hình . Ý nghĩa của tham số này là đánh giá mô hình có khả năng có bị dương tính sai không
c) Recall: số mẫu được được mô hình đánh giá đúng là dương tính chia cho số mẫu thực sự là dương tính
d) F1-score: công thức 2 * (recall * precision)/(precision + recall)
Em đánh giá trên tập test của dữ liệu (Dữ liệu có gán nhãn và chưa được đưa vào quá trình huấn luyện).

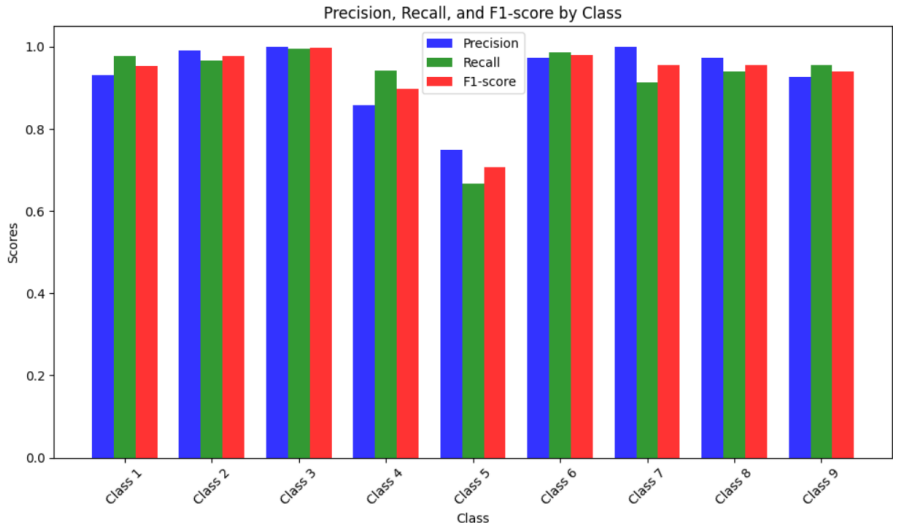
5.1 Kết quả mô hình API



Hình 7 Kết quả của mô hình lời gọi window API

Hình 7 cho thấy mô hình API phân loại các loại mã độc rất tốt trên tập test với độ chính xác là 0.97 và giá trị hàm mất mát thấp là 0.083.

Ngoài ra hình 6 cho thấy bên cạnh kết quả accuracy rất cao là 0.97, các giá trị đánh giá khác cũng rất cao. Điển hình là tham số recall và precision rất cao . Em thấy mô hình API phân loại mẫu nhãn số 2 là 100% với chỉ số precision, điều này cho thấy mô hình có khả năng bị dương tính sai rất thấp với mẫu nhãn số 2, lý do có thể là vì số lượng mẫu trong nhãn số 2 có số lượng nhiều nhất. Các chỉ số của nhãn số 4 là thấp nhất, các chỉ số precision, recall, f1-score lần lượt là 0.75, 0.67, 0.71. Điều này có thể là do số lượng mẫu của nhãn số 4 ít hơn cả trong tập dữ liệu. Điều này suy ra với số lượng mẫu đủ lớn thì có cải thiện đáng kể mô hình học sâu. Tóm lại, mô hình API có khả năng phân loại rất tốt, có tỉ lệ dương tính giả rất thấp.



Hình 8 biểu đồ phân bố điểm precision, f1-score của API

Hình 8 là biểu đồ thể hiện các tham số đánh giá của mô hình API phân loại các mã độc. Biểu đồ cho thấy các chỉ số precision, recall, f1-score là tương đối cao, trung bình là trên 0.9.

Các chỉ số này với Class 5 là thấp hơn cả, điều này em đã được phân tích ở trên là vì số mẫu trong nhãn này là thấp nhất, nên mô hình không thể học tốt được.

5.2 Kết quả mô hình Bytes

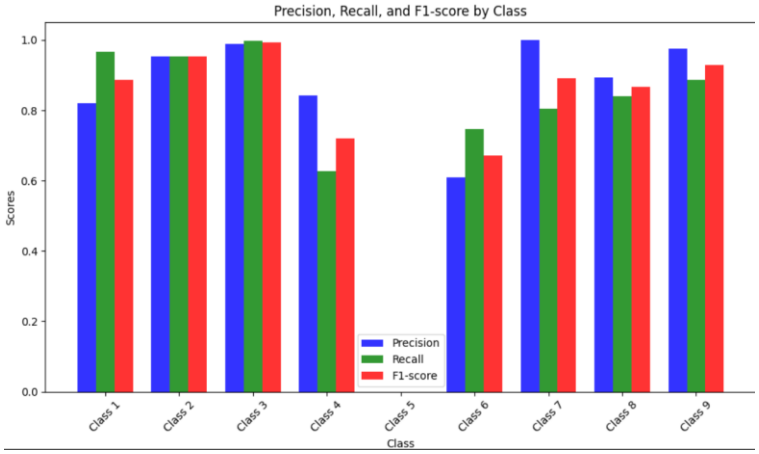
```
Test loss 0.3129643201828003; acc: 0.9123
Confusion Matrix:
 [[172   3   0   1   0   1   0   1   0]
 [  1 287   0   2   0   8   0   3   0]
 [  0   0 379   0   0   0   0   0   1]
 [  0   0   0  32   0  17   0   1   1]
 [  5   1   1   0   0   1   0   1   0]
 [ 13   0   0   1   0  56   0   5   0]
 [  1   2   2   0   0   2  37   1   1]
 [ 17   2   1   2   0   2   0 126   0]
 [  1   6   0   0   0   5   0   3 118]]
```

	precision	recall	f1-score	support
0	0.82	0.97	0.89	178
1	0.95	0.95	0.95	301
2	0.99	1.00	0.99	380
3	0.84	0.63	0.72	51
4	0.00	0.00	0.00	9
5	0.61	0.75	0.67	75
6	1.00	0.80	0.89	46
7	0.89	0.84	0.87	150
8	0.98	0.89	0.93	133
accuracy			0.91	1323
macro avg	0.79	0.76	0.77	1323
weighted avg	0.91	0.91	0.91	1323

Hình 9 Kết quả đánh giá của mô hình Bytes sequence.

Hình 9 cho thấy mô hình phân loại dựa trên đặc tính Bytes đạt kết quả accuracy khá cao là 0.91. Kết quả hàm mất mát là 0.31, cao hơn gấp 3 lần so với mô hình phân loại bằng đặc tính API.

Ngoài ra, hình 9 cũng thể hiện được bảng confusion matrix của mô hình Bytes. Nhận thấy tỉ lệ phân loại dương tính đúng của mô hình rất tốt. Hơn nữa hình 9 cho thấy nhãn số 4 có các chỉ số đều bằng 0. Như vậy, mô hình dùng đặc tính Bytes phân loại kém nhãn 4.



Hình 10 Bảng phân bố điểm precisio, recall và f1-scaore trên byte code model

Hình 10 thể hiện rõ mô hình bytes code phân loại tốt nhãn 3.

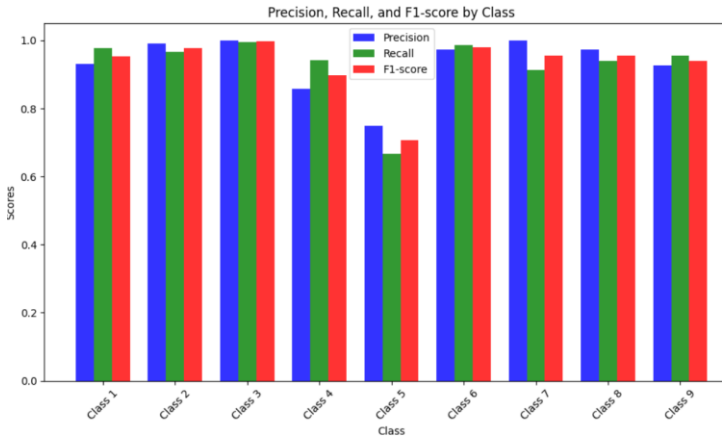
5.2 Kết quả mô hình Opcodes.

```
Test loss 0.13867677748203278; acc: 0.9682539701461792
Confusion Matrix:
 [[174   0   0   0   1   0   0   0   3]
 [  1 291   0   4   0   0   0   0   5]
 [  0   0 378   1   1   0   0   0   0]
 [  0   0   0  48   0   0   0   1   2]
 [  0   0   0   0   6   1   0   2   0]
 [  0   0   0   1   0  74   0   0   0]
 [  2   0   0   2   0   0  42   0   0]
 [  9   0   0   0   0   0   0 141   0]
 [  1   3   0   0   0   1   0   1 127]]
```

	precision	recall	f1-score	support
0	0.93	0.98	0.95	178
1	0.99	0.97	0.98	301
2	1.00	0.99	1.00	380
3	0.86	0.94	0.90	51
4	0.75	0.67	0.71	9
5	0.97	0.99	0.98	75
6	1.00	0.91	0.95	46
7	0.97	0.94	0.96	150
8	0.93	0.95	0.94	133
accuracy			0.97	1323
macro avg	0.93	0.93	0.93	1323
weighted avg	0.97	0.97	0.97	1323

Hình 11 Kết quả của mô hình opcodes

Hình 11 thể hiện mô hình opcode đạt kết quả rất cao với độ chính xác (accuracy) là 0.97 và giá trị mất mát là 0.13. Ngoài ra, mô hình bytes code cũng phân loại tốt với nhãn số 4 mặc dù số lượng mẫu nhãn số 4 là tương đối ít.



Hình 12 Bảng phân bố điểm f1-score, recall và precisio trên mô hình opcodes

5.4 Kết quả mô hình Hydra(mô hình tích hợp)

```
Test loss 0.09981387108564377; acc: 0.973
Confusion Matrix:
 [[173   2   0   0   0   0   0   2   1]
 [  0 298   0   0   0   1   0   1   1]
 [  1   1 377   0   0   0   0   1   0]
 [  0   0   0  46   0   1   0   3   1]
 [  1   0   0   0   7   0   0   0   1]
 [  1   0   0   1   0  73   0   0   0]
 [  0   1   0   0   0   0  42   2   1]
 [  8   0   0   0   0   0   0 140   2]
 [  0   0   0   0   0   0   0   1 132]]
```

	precision	recall	f1-score	support
0	0.94	0.97	0.96	178
1	0.99	0.99	0.99	301
2	1.00	0.99	1.00	380
3	0.98	0.90	0.94	51
4	1.00	0.78	0.88	9
5	0.97	0.97	0.97	75
6	1.00	0.91	0.95	46
7	0.93	0.93	0.93	150
8	0.95	0.99	0.97	133
accuracy			0.97	1323
macro avg	0.97	0.94	0.95	1323
weighted avg	0.97	0.97	0.97	1323

Hình 13 Kết quả đánh giá mô hình Hdra.

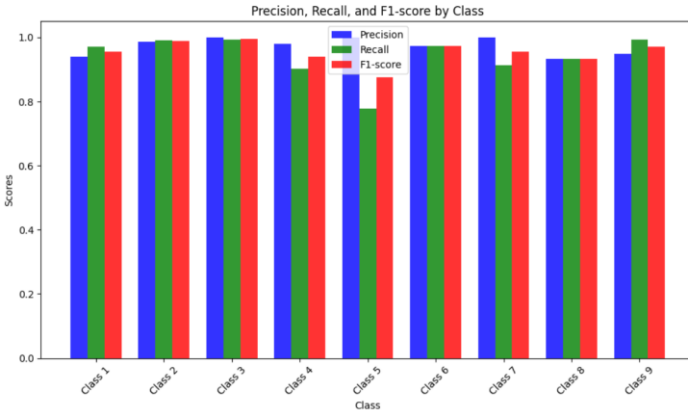
Hình 13 cho thấy mô hình tích hợp (HYDRA) đạt kết quả tốt nhất với accuracy là 0.97 và giá trị mất mát 0.09. Ngoài ra mô hình HYDRA có thể phân loại tốt các nhãn có số lượng mẫu ít , ví dụ nhãn số 4 với phân loại đúng 7 mẫu nhãn số 4 trên tổng số 9 mẫu của nhãn này trong tập test.

Bảng 1 giá trị mất mát và accuracy các mô hình

Mô hình dựa trên đặc tính	Giá trị mất mát	Accuracy	Precision trung bình	Recall trung bình	F1-score trung bình
API	0.083	0.97	0.93	0.93	0.93
Bytes	0.312	0.91	0.76	0.76	0.77
Opcodes	0.138	0.968	0.93	0.93	0.93
HYDRA	0.09	0.973	0.97	0.94	0.95

Bảng 1 thể hiện rõ sự hiệu quả của mô hình tích hợp khi so sánh với các mô hình còn lại, các điểm số đánh giá đều vượt trội. Các chỉ số Accuracy , Precision trung bình và F1-score trung bình là cao nhất và chỉ số giá trị mất mát tốt thứ 2 sau mô hình dùng đặc tính API.

Điều này cho thấy sự hiệu quả khi tạo một mô hình phân loại mã độc có kết hợp nhiều đặc tính. Điều này giúp mô hình có sự phân loại chính xác cao hơn đồng thời giảm tỉ lệ dương tính giả.



Hình 14 Bảng phân số điểm số cả mô hình Hydra

Hình 14 cho thấy các chỉ số đánh giá phân loại của mô hình HYDRA đều đạt kết quả rất tốt cho việc phân loại tất cả các nhãn mã độc.

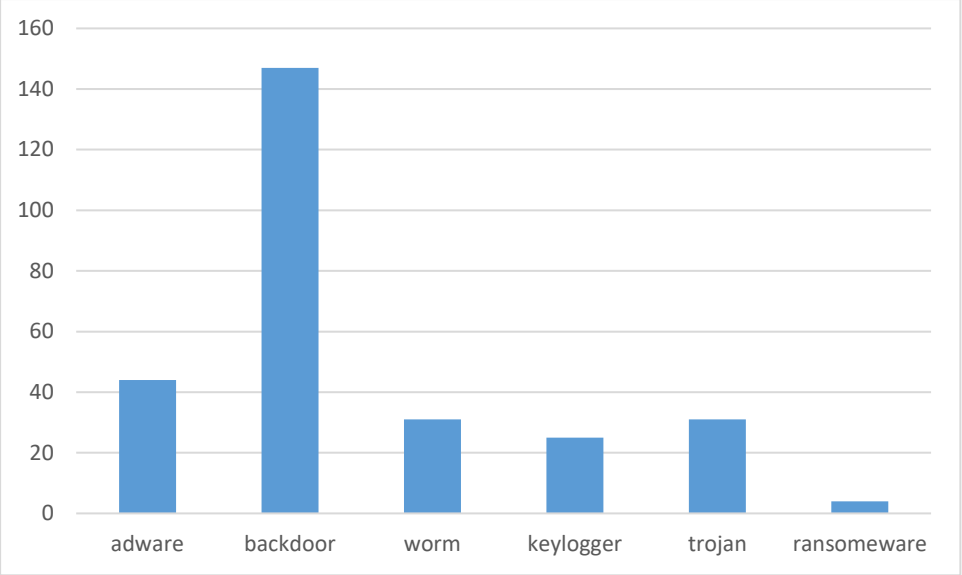
6. Thực nghiệm trên mẫu mã độc file thực thi:

Trong phần này thay vì sử dụng dataset được cung cấp bởi Microsoft Malware Classification. Em sẽ sử dụng các mẫu mã độc ở dạng tệp tin thực thi và đã được phân loại bởi trang web bazaar.abuse.ch

Các mẫu được thống kê theo bảng sau:

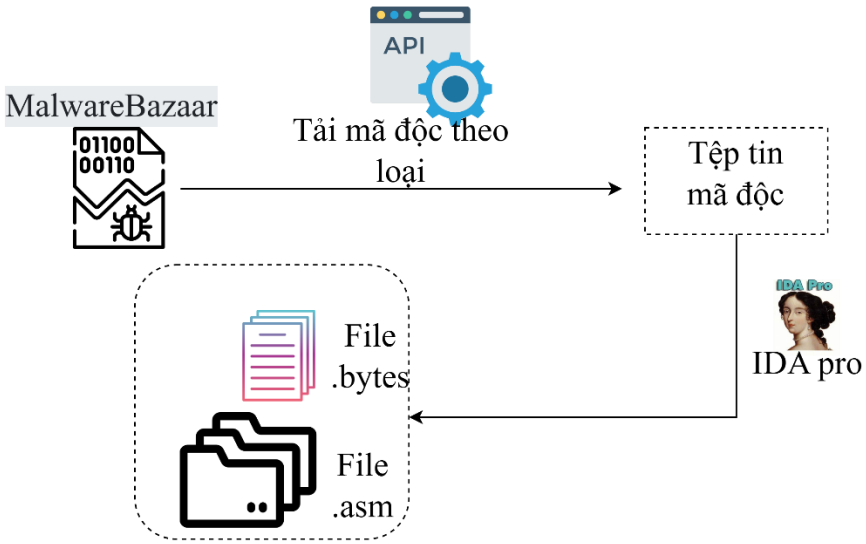
Bảng 2 Số lượng mẫu mã độc thu thập dạng tệp tin thực thi

Loại mã độc	Số lượng mẫu
Backdoor	147
Adware	44
Trojan	31
Ransomeware	4
Worm	31
keylogger	25



Hình 15 Số lượng mẫu mã độc dùng để thực nghiệm

Bảng 2 và hình 15 thể hiện số lượng mẫu cũng như phân bố số lượng các mẫu mã độc dạng tệp tin mà em đã thu thập trực tiếp từ trang web bazaar.abuse.ch. Như trong hình 15 cho thấy em thu thập tệp tin mã độc backdoor là nhiều nhất với gần 150 mẫu, và ít nhất là ransomware với 4 mẫu. Điều này cũng tương đồng về sự mất cân bằng tập dữ liệu so với mẫu dataset từ Microsoft Malware Classification.



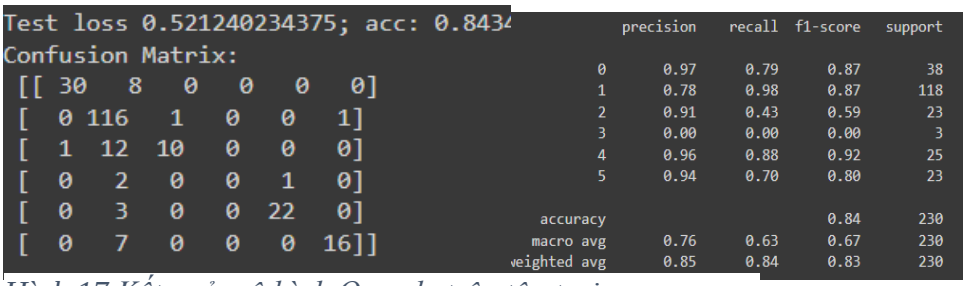
Hình 16 mô hình tổng quát thu thập dữ liệu mã độc dạng tệp tin thực thi

Hình 16 thể hiện tổng quát việc thu thập dữ liệu các mã độc dạng tệp tin thực thi và dùng IDA Pro để tạo ra dạng file là file .asm và file .bytes. Hai dạng tệp tin này được dùng để triết xuất các đặc tính là API, Bytes và mnemonic opcode.
Đặc tính bytes được triết xuất từ file .bytes và đặc tính API và mnemonic opcode được triết xuất từ file .asm.

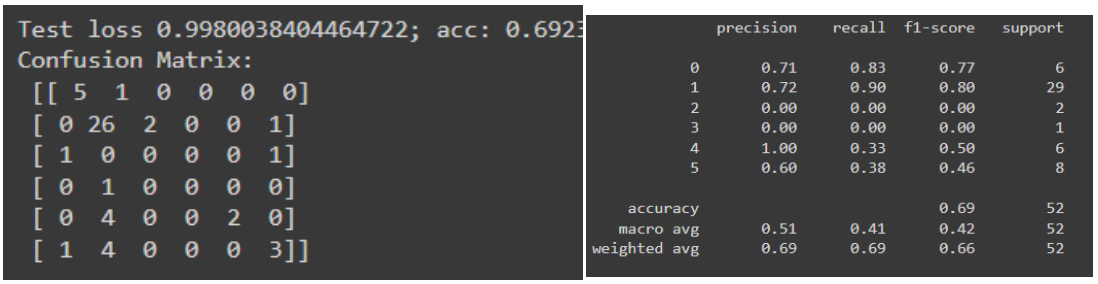
Sau đó, em dùng các phương pháp xử lý như trong bài báo để triết xuất các đặc tính của mã độc. Em cũng dùng mô hình học máy như trong bài báo để huấn luyện. Cụ thể, em sẽ huấn luyện 3 mô hình nhánh con của từng đặc tính là chuỗi bytecode, chuỗi các mnemonic opcode, và API. Sau đó, em huấn luyện mô hình HYDRA kết hợp ba mô hình lại với nhau.

6.1 Kết quả thực nghiệm:

Mô hình Opcode



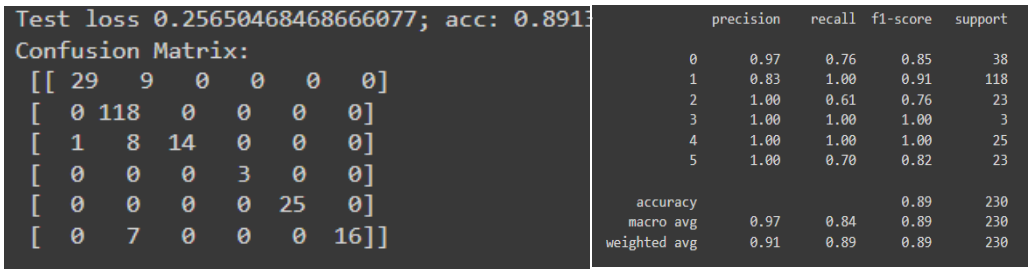
Hình 17 Kết quả mô hình Opcode trên tập train



Hình 18 Kết quả mô hình opcode trên tập test

Hình 17 và hình 18 cho biết kết quả của mô hình sử dụng đặc tính opcode. Nhận thấy mô hình phân loại tốt nhất đối với nhãn 1(backdoor) trong cả tập train và test. Nguyên nhân có thể là do số lượng mẫu trong nhãn này là lớn nhất.

Mô hình API

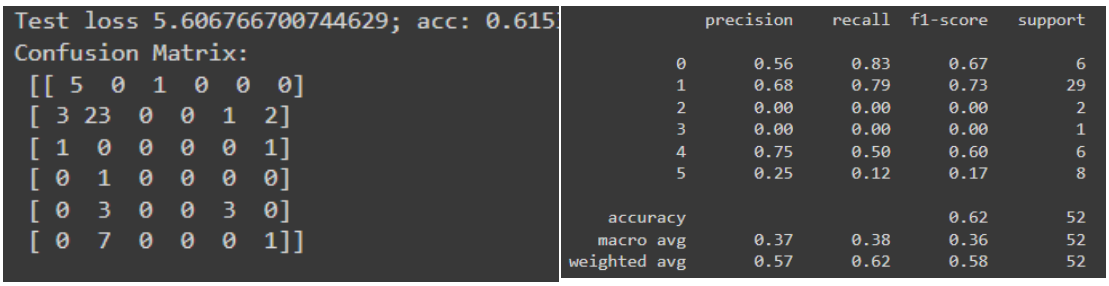


```
Test loss 0.25650468468666077; acc: 0.891
Confusion Matrix:
 [[ 29   9   0   0   0   0]
 [  0 118   0   0   0   0]
 [  1   8  14   0   0   0]
 [  0   0   0   3   0   0]
 [  0   0   0   0  25   0]
 [  0   7   0   0   0  16]]
```

	precision	recall	f1-score	support
0	0.97	0.76	0.85	38
1	0.83	1.00	0.91	118
2	1.00	0.61	0.76	23
3	1.00	1.00	1.00	3
4	1.00	1.00	1.00	25
5	1.00	0.70	0.82	23
accuracy			0.89	230
macro avg	0.97	0.84	0.89	230
weighted avg	0.91	0.89	0.89	230

Hình 19 Kết quả mô hình API trên tập train

Hình 19 cho thấy kết quả trên tập train tốt hơn so với mô hình opcode.



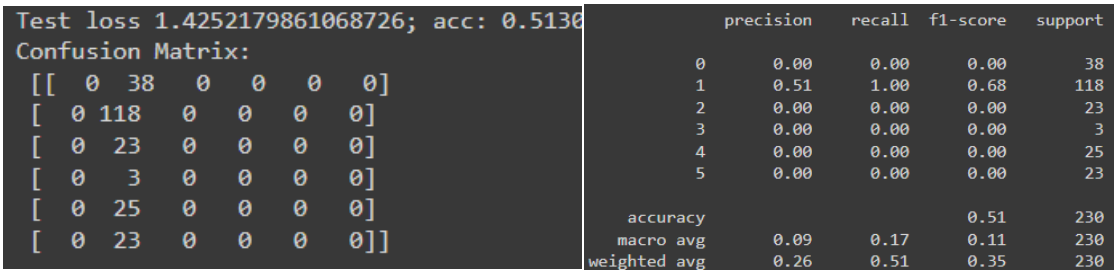
```
Test loss 5.606766700744629; acc: 0.615
Confusion Matrix:
 [[ 5  0  1  0  0  0]
 [ 3 23  0  0  1  2]
 [ 1  0  0  0  0  1]
 [ 0  1  0  0  0  0]
 [ 0  3  0  0  3  0]
 [ 0  7  0  0  0  1]]
```

	precision	recall	f1-score	support
0	0.56	0.83	0.67	6
1	0.68	0.79	0.73	29
2	0.00	0.00	0.00	2
3	0.00	0.00	0.00	1
4	0.75	0.50	0.60	6
5	0.25	0.12	0.17	8
accuracy			0.62	52
macro avg	0.37	0.38	0.36	52
weighted avg	0.57	0.62	0.58	52

Hình 20 Mô hình API trên tập test

Tương tự như mô hình opcode nhãn số 1 được phân loại tốt nhất.

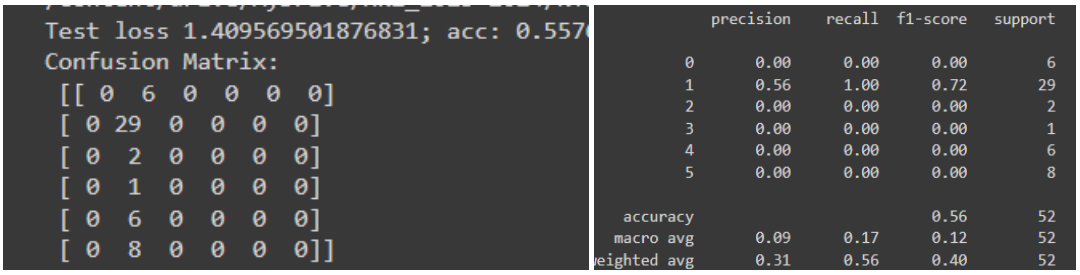
Mô hình Bytes



```
Test loss 1.4252179861068726; acc: 0.5130
Confusion Matrix:
 [[  0  38   0   0   0   0]
 [  0 118   0   0   0   0]
 [  0  23   0   0   0   0]
 [  0   3   0   0   0   0]
 [  0  25   0   0   0   0]
 [  0  23   0   0   0   0]]
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	38
1	0.51	1.00	0.68	118
2	0.00	0.00	0.00	23
3	0.00	0.00	0.00	3
4	0.00	0.00	0.00	25
5	0.00	0.00	0.00	23
accuracy			0.51	230
macro avg	0.09	0.17	0.11	230
weighted avg	0.26	0.51	0.35	230

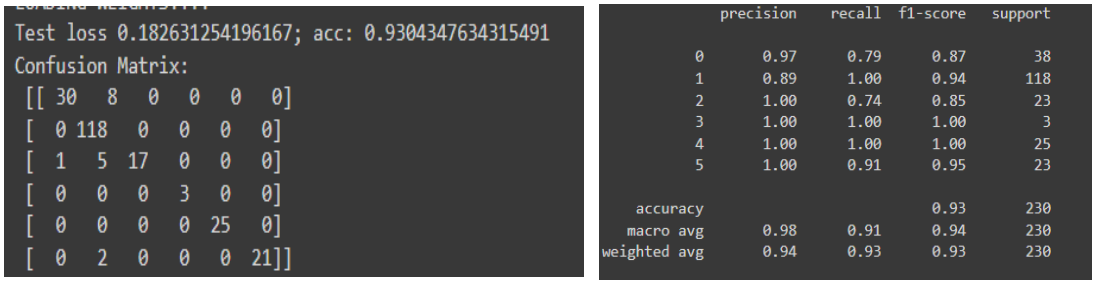
Hình 21 Kết quả trên tập train của mô hình Bytes code



Hình 22 Kết quả trên tập test của mô hình bytecode

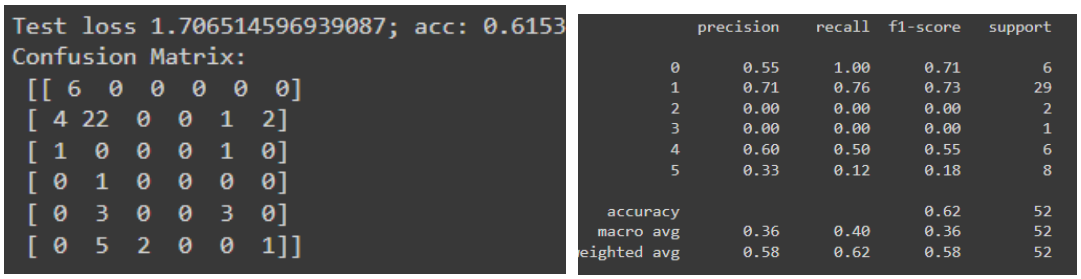
Hình 21 và 22 cho thấy kết quả accuracy trên cả tập train và test của mô hình bytes code đều rất thấp đều dưới 0.6.

Mô hình Hydra



Hình 23 kết quả trêm tập train của mô hình Hydra

- Hình 23 cho thấy kết quả trên tập train của mô hình HYDRA là tốt nhất trong các mô hình, với accuracy là 0.93 và giá trị mất mát là 0.18



Hình 24 kết quả trên tập test của mô hình HYDRA

Hình 24 cho thấy kết quả trên tập test thì accuracy còn thấp của mô hình HYDRA.

Bảng 3 Kết quả đáng giá các mô hình trên tập train

Mô hình đặc tính	Giá trị mất mát	Accuracy	Precision	Recall	F1-score
Opcodes	0.52	0.84	0.76	0.63	0.67
API	0.25	0.89	0.97	0.84	0.89
Bytes	1.4	0.5	0.09	0.17	0.11
HYDRA	0.18	0.93	0.98	0.91	0.94

Bảng 3 cho biết kết quả đánh giá các mô hình trên tập train.

Bảng 4 Kết quả đánh giá mô hình trên tập test

Mô hình đặc tính	Giá trị mất mát	Accuracy	Precision	Recall	F1-score
Opcodes	0.99	0.69	0.51	0.41	0.42
API	5.6	0.615	0.37	0.38	0.36
Bytes	1.4	0.557	0.09	0.17	0.12
HYDRA	1.7	0.615	0.36	0.4	0.36

Bảng 4 cho biết kết quả đánh giá các mô hình trên tập test

7. So sánh kết quả thực nghiệm của em và của tác giả bài báo

Bảng 5 Kết quả huấn luyện mô hình học sâu của nhóm em.

Mô hình dựa trên đặc tính	Giá trị mất mát	Accuracy	Precision trung bình	Recall trung bình	F1-score trung bình

API	0.083	0.97	0.93	0.93	0.93
Bytes	0.312	0.91	0.76	0.76	0.77
Opcodes	0.138	0.968	0.93	0.93	0.93
HYDRA	0.09	**0.973**	0.97	0.94	**0.95**

Bảng 6 Kết quả đánh giá mô hình học sâu của tác giả

Model	Accuracy	Macro F1-score
API-based Feedforward Network	0.9833	0.9621
Assembly-based Shallow CNN	0.9917	0.9856
Bytes-based DeepConv	0.9756	0.8902
HYDRA	0.9871	0.9695
HYDRA (Pretraining, Modality Dropout)	**0.9975**	**0.9954**

Hình 5 và hình 6 thể hiện kết quả đánh giá các mô hình của em và của tác giả. Quan sát hai hình này cho thấy ở mô hình HYDRA kết quả Accuracy và F1-score của em đạt gần bằng của tác giả.

8. Kết luận

Thông qua nghiên cứu bài báo "HYDRA: A multimodal deep learning framework for malware classification" em đã đạt kết quả như sau:

Em đã thực nghiệm lại nghiên cứu của tác giả và kết quả là đạt gần giống với bài báo đặc biệt là ở mô hình HYDRA .

Em đã thực nghiệm lại trên tập dữ các file mã độc PE (thu thập trong phần 6), bao gồm tự thu thập các mã độc đã gán nhãn, triết xuất các đặc trưng, tiền xử lý và ứng dụng mô hình sâu để phân loại chúng.

9. Công việc tương lai.

Trong tương lai em sẽ sử dụng trên việc phân loại các loại mã độc khác nhau trên các nguồn khác nhau.
Sử dụng thêm nhiều mô hình để đánh giá.
Sử dụng thêm các đặc trưng khác.

10. Link sản phẩm và video demo:

- Link đường dẫn chứa toàn bộ code và colab notebook và các data được triết xuất của nhóm em: https://drive.google.com/drive/folders/1_86OHLXP4-WkbGf1VDpIN_5G4q79Un4c?usp=sharing

- Link Dataset: https://drive.google.com/drive/folders/1odwmoRnPylbFM0ge1OvLgjQ-gN-ZmusA?usp=sharing

- Video Demo kết quả trên tập test học sâu: DEMO đồ án Cơ chế mã độc.webm

- Demo cách triết xuất các đặc tính từ tệp tin mã độc thực thi

DEMO triết xuất đặc tính từ tệp tin mã độc .webm

11. Tài liệu tham khảo

[1] Gibert, D., et al. (2020). "HYDRA: A multimodal deep learning framework for malware classification." Computers & Security **95**: 101873.

[2] *Introduction* (no date) *Home - PE Parser Documentation*. Available at: https://pe-parser.readthedocs.io/en/latest/ (Accessed: 22 May 2024).

HẾT