



BÁO CÁO THỰC HÀNH

Môn học: Quản trị mạng và hệ thống

Lab 3: Advanced Malware Detection

GVHD: Nguyễn Hữu Quyền

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT522.021.ATCL.2- Nhóm 3

STT	Họ và tên	MSSV	Email
1	Nguyễn Ngọc Trà My	21520353	21520353@gm.uit.edu.vn
2	Bùi Hoàng Trúc Anh	21521817	21521817@gm.uit.edu.vn
3	Lê Hoàng Oanh	21521253	21521253@gm.uit.edu.vn
4	Huỳnh Minh Tân Tiến	21521520	21521520@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:

STT	Công việc	Kết quả tự đánh giá
1	Yêu cầu 1	100%
2	Yêu cầu 2	100%
3	Yêu cầu 3	100%
4	Yêu cầu 4	100%
5	Yêu cầu 5	100%
6	Yêu cầu 6	100%
7	Yêu cầu 7	100%
8	Yêu cầu 8	100%

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

BÁO CÁO CHI TIẾT

1. Cho biết kết quả accuracy và confusion matrix.

Bước 1: Import các thư viện cần thiết để xử lý nội dung JavaScript, chuẩn bị tập dữ liệu, phân loại và đo hiệu suất bộ phân loại.

```
import os
from sklearn.feature_extraction.text import HashingVectorizer, TfidfTransformer
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.pipeline import Pipeline
```

Bước 2: Định nghĩa đường dẫn 2 thư mục Javascripts đã giải nén.

```
js_path = "/content/drive/MyDrive/lab_dataset/lab3/JavascriptSamplesNotObfuscated"
obfuscated_js_path = "/content/drive/MyDrive/lab_dataset/lab3/JavascriptSamplesObfuscated"

corpus = []
labels = []

file_types_and_labels = [(js_path, 0), (obfuscated_js_path, 1)]
```

Bước 3: Tiếp theo ta sẽ gán nhãn cho chúng.

```
for files_path, label in file_types_and_labels:
    files = os.listdir(files_path)
    for file in files:
        file_path = files_path + "/" + file
        try:
            with open(file_path, "r") as myfile:
                data = myfile.read().replace("\n", "")
                data = str(data)
                corpus.append(data)
                labels.append(label)
        except:
            pass
```

Bước 4: Ta chia tập dữ liệu thành tập huấn luyện và tập thử nghiệm, đồng thời tạo pipeline cho NLP, tiếp theo sử dụng phân loại random forest.

```
X_train, X_test, y_train, y_test = train_test_split(
    corpus, labels, test_size=0.33, random_state=42
)
text_clf = Pipeline(
    [
        ("vect", HashingVectorizer(input="content", ngram_range=(1,
3))),
        ("tfidf", TfidfTransformer(use_idf=True,)),
        ("rf", RandomForestClassifier(class_weight="balanced")),
    ]
)
```

Bước 5: Sau đó chạy huấn luyện và cho ra đánh giá.

```
text_clf.fit(X_train, y_train)
y_test_pred = text_clf.predict(X_test)
print("accuracy: " + str(accuracy_score(y_test, y_test_pred)))
print(confusion_matrix(y_test, y_test_pred))
```

Kết quả:

```
✓ 52s [53] text_clf.fit(X_train, y_train)
y_test_pred = text_clf.predict(X_test)
print("accuracy: " + str(accuracy_score(y_test, y_test_pred)))
print(confusion_matrix(y_test, y_test_pred))

accuracy: 0.9775583482944344
[[620 15]
 [ 10 469]]
```

2. Cho biết kết quả vector X.

Bước 1: Import IPython để thu thập các output của script.

```
✓ 0s from IPython.utils import io
from os import listdir
```

Bước 2: Định nghĩa hàm trích xuất thuộc tính. Chạy pdfid đọc một tập và lấy kết quả output của chúng. Kế tiếp, phân tích output để lấy vector số.

```
[2] def PDF_to_FV(file_path,pdfid_path):  
    """Featurize a PDF file using pdfid."""  
    with io.capture_output() as captured:  
        %run -i pdfid $file_path  
    out = captured.stdout  
    out1 = out.split("\n")[2:-2]  
    return [int(x.split()[-1]) for x in out1]
```

Bước 3: Import listdir để liệt kê các tập tin của thư mục PDF

```
PDFs_path = "/content/drive/MyDrive/lab_dataset/lab3/PDFSamples"  
pdfid_path = "/content/drive/MyDrive/lab_dataset/lab3/pdfid.py"  
print(pdfid_path)
```

```
/content/drive/MyDrive/lab_dataset/lab3/pdfid.py
```

Bước 4: Cho vào vòng lặp để trích xuất, quét hết tất cả tập tin vào mảng X.

```
X = []  
files = listdir(PDFs_path)  
for file in files:  
    file_path = PDFs_path + "/" + file  
    X.append(PDF_to_FV(file_path, pdfid_path))  
  
for vector in X:  
    print(vector)
```

Kết quả

```
X = []
files = listdir(PDFs_path)
for file in files:
    file_path = PDFs_path + "/" + file
    X.append(PDF_to_FV(file_path, pdfid_path))

for vector in X:
    print(vector)

[1096, 1095, 1061, 1061, 0, 0, 2, 32, 0, 43, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0]
[153, 153, 82, 82, 2, 2, 2, 7, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0]
```

3. Cho biết kết quả vector X.

Xem kết quả chi tiết tại file Notebook (.ipynb)

4. Cho biết kết quả đánh giá.

Xem kết quả chi tiết tại file Notebook (.ipynb)

5. Cho biết kết quả đánh giá mô hình qua tập test.

```
[ ] print(model.evaluate(X, Y))

9/9 [=====] - 3s 248ms/step - loss: 0.4851 - acc: 0.8524
[0.4850543737411499, 0.8523985147476196]
```

6. Cài đặt. UPX từ <https://github.com/1upx/upx/releases>, và tiến hành đóng gói các tập tin pe tại Benign PE Samples UPX

```
import os
files_path = "/content/drive/MyDrive/ML/Benign PE Samples UPX"
files = os.listdir(files_path)
file_paths = [files_path+x for x in files]







from subprocess import Popen, PIPE
cmd = "upx.exe"
for path in file_paths:
    cmd2 = cmd + " \"" + path + "\""
    res = Popen(cmd2, stdout=PIPE).communicate()
    print(res)
    if "error" in str(res[0]):
        print(path)
        os.remove(path)
```

Kết quả chạy khi packed thành công:

8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4418880 ->	1028352	41.45%	win32/pe	appidtel.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4406784 ->	1023744	41.38%	win32/pe	ARP.EXE\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4423168 ->	1041664	41.64%	win32/pe	aspnet_regiis.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4423168 ->	1041664	41.64%	win32/pe	aspnet_state.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4423168 ->	1041664	41.64%	win32/pe	aspnet_wp.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4414976 ->	1032448	41.51%	win32/pe	at.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4435456 ->	1055488	41.83%	win32/pe	AtBroker.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4418880 ->	1028352	41.45%	win32/pe	attrib.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4419072 ->	1037856	41.57%	win32/pe	auditpol.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4308592 ->	1014528	41.25%	win32/pe	BackgroundTaskHost.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4427264 ->	1046272	41.70%	win32/pe	BackgroundTransferHost.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4431360 ->	1050880	41.77%	win32/pe	bthudtask.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4435456 ->	1055488	41.83%	win32/pe	ByteCodeGenerator.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4414976 ->	1032448	41.51%	win32/pe	calcis.exe\n\nPacked 1 file.\n', None)
8th 2022\n\n	File size	Ratio	Format	Name\n	-----\n	-----\n	-----\n	4414976 ->	1032448	41.51%	win32/pe	calc.exe\n\nPacked 1 file.\n', None)

7. Cho biết kết quả đánh giá.

Giải nén những folder Benign có từ bài lab 2 và cho vô 1 thư mục chung là Benign PE Samples

-  Benign PE Samples 1
-  Benign PE Samples 2
-  Benign PE Samples 3
-  Benign PE Samples 4
-  Benign PE Samples 5
-  Benign PE Samples 6

Bước 1: Đọc tất cả tập tin cần phân tích và gán nhãn cho chúng.

```
import os
from os import listdir

directories_with_labels = [
    ("/content/drive/MyDrive/Lab3/Benign PE Samples", 0),
    ("/content/drive/MyDrive/Lab3/Benign PE Samples UPX", 1),
    ("/content/drive/MyDrive/Lab3/Benign PE Samples Amber", 2),
]

list_of_samples = []
labels = []
for dataset_path, label in directories_with_labels:
    samples = [f for f in listdir(dataset_path)]
    for file in samples:
        file_path = os.path.join(dataset_path, file)
        list_of_samples.append(file_path)
        labels.append(label)
```

Bước 2: Phân ra train test.

```
[3] from sklearn.model_selection import train_test_split

samples_train, samples_test, labels_train, labels_test = train_test_split(
    list_of_samples,
    labels,
    test_size=0.3,
    stratify=labels,
    random_state=11
)
```

Bước 3: Import thư viện cần thiết để trích xuất N-grams.

```
import collections
from nltk import ngrams
import numpy as np
```

Bước 4: Định nghĩa hàm sử dụng trích xuất N-grams.

```

0s def read_file(file_path):
    """Reads in the binary sequence of a binary file."""
    with open(file_path, "rb") as binary_file:
        data = binary_file.read()
    return data

def byte_sequence_to_Ngrams(byte_sequence, N):
    """Creates a list of N-grams from a byte sequence."""
    Ngrams = ngrams(byte_sequence, N)
    return list(Ngrams)

def extract_Ngram_counts(file, N):
    """Takes a binary file and outputs the N-grams counts of its binary
    sequence."""
    filebyte_sequence = read_file(file)
    file_Ngrams = byte_sequence_to_Ngrams(filebyte_sequence, N)
    return collections.Counter(file_Ngrams)

def featurize_sample(sample, K1_most_frequent_Ngrams_list):
    """Takes a sample and produces a feature vector.
    The features are the counts of the K1 N-grams we've selected.
    """
    K1 = len(K1_most_frequent_Ngrams_list)
    feature_vector = K1 * [0]
    file_Ngrams = extract_Ngram_counts(sample, N)
    for i in range(K1):
        feature_vector[i] = file_Ngrams[K1_most_frequent_Ngrams_list[i]]
    return feature_vector

```

Bước 5: Chọn N-grams mong muốn.

```

3m [6] N = 2
    total_Ngram_count = collections.Counter({})
    for file in samples_train:
        total_Ngram_count += extract_Ngram_counts(file, N)
    K1 = 100
    K1_most_common_Ngrams = total_Ngram_count.most_common(K1)
    K1_most_common_Ngrams_list = [x[0] for x in K1_most_common_Ngrams]

```

Bước 6: Thiết lập thuộc tính để huấn luyện.

```

1m def main():
    Ngram_features_list_train = []
    y_train = []
    for i in range(len(samples_train)):
        file = samples_train[i]
        Ngram_features = featurize_sample(file, K1_most_common_Ngrams_list)
        Ngram_features_list_train.append(Ngram_features)
        y_train.append(labels_train[i])
    X_train = Ngram_features_list_train

```

Bước 7: Huấn luyện mô hình random forest trên tập train.

```

0s from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(n_estimators=100)
clf = clf.fit(X_train, y_train)

```

Bước 8: Thiết lập thuộc tính cho tập test.

```

1m 1m ▶ Ngram_features_list_test = []
      y_test = []
      for i in range(len(samples_test)):
          file = samples_test[i]
          Ngram_features = featurize_sample(file, K1_most_common_Ngrams_list)
          Ngram_features_list_test.append(Ngram_features)
          y_test.append(labels_test[i])
      X_test = Ngram_features_list_test

```

Bước 9: Sử dụng bộ phân loại được đào tạo để dự đoán trên bộ test và đánh giá hiệu suất bằng confusion matrix.

```

0s 0s ▶ y_pred = clf.predict(X_test)

      from sklearn.metrics import confusion_matrix
      confusion_matrix(y_test, y_pred)

array([[113,  0,  0],
       [  0, 60,  0],
       [  0,  0, 23]])

```

8. Cho biết kết quả đánh giá mẫu mới trong việc đánh lừa bộ nhận diện.

Bước 1: Thêm các thư viện.

```

0s [8] import os
      import pandas as pd
      from keras.models import load_model

```

Sửa đổi file python trong thư mục MalGAN trước khi import

- File MalGAN_utils.py

```

1 1 import numpy as np
2 2 import pandas as pd
3 3 import tensorflow as tf
4 4 from tensorflow.compat.v1.keras.backend import set_session
5 5 from MalGAN_preprocess import preprocess

```

```

31 31 def limit_gpu_memory(per):
32 32     config = tf.compat.v1.ConfigProto()
33 33     config.gpu_options.per_process_gpu_memory_fraction = per
34 34     set_session(tf.compat.v1.Session(config=config))
35 35

```

- File MalGAN_preprocess.py


```
1 import os
2 import time
3 import pickle
4 import argparse
5 import pandas as pd
6 import tensorflow as tf
7 from tensorflow.keras.utils import pad_sequences
```

Phải di chuyển tới thư mục MalGAN trước khi import thư viện

```
%cd "/content/drive/MyDrive/Lab3/MalGAN"
```

```
/content/drive/MyDrive/Lab3/MalGAN
```

Import MalGAN_gen_adv_examples và MalGAN_utils

```
[18] import MalGAN_gen_adv_examples
import MalGAN_utils
```

Bước 2: Xác định các đường dẫn dữ liệu.

```
save_path = "/content/drive/MyDrive/Lab3/MalGAN/MalGAN_output"
model_path = "/content/drive/MyDrive/Lab3/MalGAN/MalGAN_input/malconv.h5"
log_path = "/content/drive/MyDrive/Lab3/MalGAN/MalGAN_output/adversarial_log.csv"
pad_percent = 0.1
threshold = 0.6
step_size = 0.01
limit = 0.
input_samples = "/content/drive/MyDrive/Lab3/MalGAN/MalGAN_input/samplesIn.csv"
```

Bước 3: Sử dụng GPU.

```
MalGAN_utils.limit_gpu_memory(limit)
```

Bước 4: Đọc tập tin csv chứa name và label của mẫu vào dataframe.

```
df = pd.read_csv(input_samples, header=None)
fn_list = df[0].values
```

Bước 5: Tải mô hình MalConv đã được huấn luyện.

```
model = load_model(model_path)
```

Bước 6: Sử dụng Fast Gradient Step Method (FGSM) để tạo mẫu đối kháng.

```

0s 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000

adv_samples, log = MalGAN_gen_adv_examples.gen_adv_samples(
    model,
    fn_list,
    pad_percent,
    step_size,
    threshold)

0778a070b283d5f4057aeb3b42d58b82ed20e4eb_f205bd9628ff8dd7d99771f13422a665a70bb916 not exist

ValueError                                Traceback (most recent call last)
<ipython-input-29-3fd5446e8c57> in <cell line: 1>()
----> 1 adv_samples, log = MalGAN_gen_adv_examples.gen_adv_samples(
      2     model,
      3     fn_list,
      4     pad_percent,
      5     step_size,

3 frames
/content/drive/MyDrive/Lab3/MalGAN/MalGAN_gen_adv_examples.py in gen_adv_samples(model, fn_list, pad_percent, step_size, thres)
33     """ run one file at a time due to different padding length, [slow]
34     inp, len_list = preprocess([fn], max_len)
---> 35     inp_emb = np.squeeze(np.array(inp2emb([inp, False])), 0)
36
37     pad_idx = len_list[0]

/usr/local/lib/python3.10/dist-packages/keras/src/backend.py in func(model_inputs)
4657
4658     def func(model_inputs):
-> 4659         outs = model(model_inputs)
4660         if wrap_outputs:
4661             outs = [outs]

/usr/local/lib/python3.10/dist-packages/keras/src/utils/traceback_utils.py in error_handler(*args, **kwargs)
68     # To get the full stack trace, call:
69     # `tf.debugging.disable_traceback_filtering()`
---> 70     raise e.with_traceback(filtered_tb) from None
71     finally:
72         del filtered_tb

/usr/local/lib/python3.10/dist-packages/keras/src/engine/input_spec.py in assert_input_compatibility(input_spec, inputs, layer_name)
217
218     if len(inputs) != len(input_spec):
-> 219         raise ValueError(
220             f'Layer "{layer_name}" expects {len(input_spec)} input(s), '
221             f"but it received {len(inputs)} input tensors. ")

ValueError: Layer "model_2" expects 1 input(s), but it received 2 input tensors. Inputs received: [<tf.Tensor: shape=(0, 80000), dtype=int32, numpy=array([], shape=(0, 80000), dtype=int32)>, <tf.Tensor: shape=(), dtype=bool, numpy=False>]

```

Bước 7. Lưu lại log và ghi mẫu mới vào thư mục.