Information Retrieval System for Malwares

Anonymous Author(s)

ABSTRACT

In this paper, we describe the problems that arise when building a malware retrieval system, and how to solve them. Multilabel Weighted Centerloss(MWC) is proposed to learn semantic distances between malware samples. Also we visualize the embedding vectors of malware samples and show the retrieval results to prove the synchronization between our perception of malwares and learned embedding space. [**?**]

CCS CONCEPTS

• **Security and privacy → Malware and its mitigation**;

KEYWORDS

malware; information retrieval

1 INTRODUCTION

하루에 30만개 이상의 멀웨어가 생성되고 있고 이 숫자는 계속 늘어나고 있다. 같은 기능을 하지만 해시는 다른 멀웨어들을 자동으로 제너레이션 하는 멀웨어 obfuscation 툴들의 발전 및 배포가 그 이유이다. 반면 보안 전문가의 시간은 늘어나지 않고 있다. 보안 전문가는 크게 두 가지 일을 하는데, 하나는 해시로 걸러지지 않은 악성코드의 카테고리를 분류하는 일이고 두 번째는 주요 악성코드에 deep dive 해서 obfuscation 방법이나 악성 행위 방법을 공부하고 후에 나올 악성코드에 잘 대응할 수 있게 하는 것이다. 이 두 가지 주요 업무 중에서 머신러닝이 쉽게 도움을 줄 수 있는 부분은 두 번째 태스크이다. 사실상 수많은 new 악성코드들을 보안 전문가들이 하나하나 정적 혹은 동적 분석을 통해 분류하는 일은 불가능하다. 따라서 자동으로 멀웨어를 분류하는 시스템은 예로부터 많이 만들었다. PE 스트럭쳐나 콜그래프 등의 정적 피쳐를 받아서 SVM, RF, DNN 등으로 멀웨어를 분류하는 모델이 연구되어왔다. 하지만 이렇게 만들어진 분류기들에는 문제가 있다. 그 시스템들은 학습된 샘플들에 대해서는 좋은 정확도를 보이지만 새로운 샘플에 대한 대응이 좋지 못하다. 소분류를 레이블로 주고 가르치면 종류가 너무 많고 계속 추가되는 소분류가 지속해서 생긴다. 대분류를 레이블로 주고 가르치면 정적 피쳐 기준 이너클래스 베리언트가 너무 커지고 레이블의 모호성이 발생한다. 따라서 보안 전문가는 멀웨어 분류를 위해 학습된 머신러닝 모델을 사용하고나서 그 결과를 확인해야하는 수고를 해야한다. 그리고 그 확인 과정은 비슷한 멀웨어들을 검색해주는 시스템이 구축되어있지 않다면 오래 걸리게 된다. 따라서 classification model 을 검증하는데에 사용할 수 있으면서도 동시에 위에서 언급한 보안 전문가의 두번째 태스크인 deep dive 에도 도움을 줄 수 있는 멀웨어 IR System 에 대한 니즈가 증가하고 있다. 기존 멀웨어 IR system 은 크게 LSH나 imphash와 같은 hand-craft Feature representation 을 사용하는 시스템과, ML을 트레이닝시키며 얻은 automatically extracted feature representation 을 사용하는 시스템이 있다. 전자의 장점은 False Negative 가 낮다는 것이지만, generalization 이 잘 안된다는 단점이 있다.[] 더 좋은 generalization 을 위해 후자의 기법들이 개발되었지만, 멀웨어 representation location 이 사람의 시멘틱 스페이스와 어긋나있다는 단점이 IR 결과의 정상 평가 결과(?)를 낮추고 있다. 우리는 이러한 문제들을 해결하고 사람의 인지스페이스와 멀웨어 임베딩 스페이스가 동기화되어있는 멀웨어 IR 시스템을 제안한다. 또한 우리는 멀웨어 인포메이션 리트리벌 시스템을 만들 때의 어려운 점들과 그 문제들을 어떻게 해결했는지를 설명한다. 그리고 쿼링 결과를 시각적으로 보여서 우리의 인지와 임베딩 스페이스가 얼마나 동기화되어있는지 확인한다. 페이퍼 구성은 다음과 같다: related work 가 섹션 2에 위치한다. 섹션 3은 우리가 풀고자하는 문제를 정의하고 어떤 IR 시스템이 좋은 시스템인지에 대한 이야기를 한다. 섹션 4은 우리가 제안하는 malware IR 시스템의 디자인과 구현 과정을 설명한다. 구현 과정에는 크게 두 페이즈가 있는데 첫 째는 섹션 5에서 설명할 임베딩 학습 페이즈이고 둘 째는 섹션 5에서 설명할 리트리벌 페이즈이다. 섹션 6에서는 실험에 쓰인 뉴럴넷과 데이터셋,그리고 실험 결과에 대해서 설명하고 임베딩 벡터를 시각화한다. 섹션 7에서는 Future works 그리고 세션 8 에서는 결과를 보여준다.

2 RELATED WORKS

Nataraj et al. 은 large scale malware search and retrieval system 을 제안하였다. 이 논문에서는 malware image로 부터 fingerprints 를 얻고, 이를 nearest neighbor search를 통해 비슷한 샘플을 retrieval하는 방법을 제안하였다. Upchurch et al. 은 similarity testing을 통해 variant malware를 탐지하는 framework를 소개했다. 이 framework는 BitShred, TLSH, sdhash, ssdeep 등의 방법으로 정적 feature를 추출하고 이를 비교함으로써 유사한 malware인지의 여부를 판단하였다. Palahan et al. 은 system call dependency graph 로부터 significant malicious behaviors를 추출하고 이를 비교함으로써 malware간의 similarity를 비교하는 방법을 제시했다. Neural IR Multilabel embedding

3 PROBLEM SETTINGS

우리가 제안하는 malware IR 시스템은 벡터 학습 과 리트리벌 두 페이즈로 나뉜다. 두 페이즈에서 사용될 노테이션들을 정의하고, 이들을 이용해서 각 페이즈에서 어떤 태스크를 행해야하는지를 설명한다.

3.1 Notations

우리는 먼저 학습 멀웨어 셋으로 Y를 갖고 있다. 그리고 각 멀웨어에 해당하는 싱글레이블 셋 Ys 과 멀티레이블 셋 Ym 이 있다. 이 레이블을 얻는 방법은 섹션 6.1에서 자세히 설명한다. 그리고 각 멀웨어로부터 손으로 추출된 피쳐의 튜플들의 셋을 V 로 정의한다. 어떤 피쳐들을 추출해서 사용했는지는 섹션 6.2에서 설명한다. 멀웨어 IR 시스템은 4개의 원소를 갖는 tuple 로 정의된다. h 는 멀웨어 피쳐로부터 벡터 representation 을 얻을 수 있는 임베더 함수이다. E 는 검색이 될 멀웨어의 임베딩 벡터 셋이다. d 는 입력된 쿼리와 임베딩 벡터 간 거리를 계산하는 함수이다. R 은 리트리벌 결과를 반환하는 프레임워크이다. 학습 페이즈에서는 적절한 h 를 학습하고 이를 통해 Z로부터 E를 얻는다. 리트리벌 페이즈에서는 멀웨어 샘플 쿼리 q 를 입력받고 가장 가까운 k 개의 neighbor 를 랭킹 모듈 R 을 통해 반환한다.

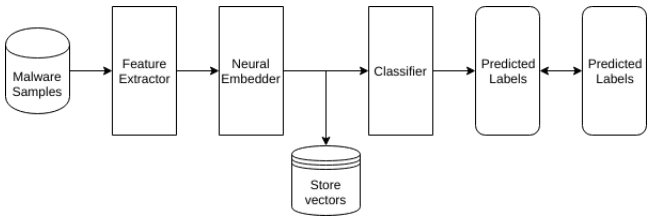

Figure 1: train phase

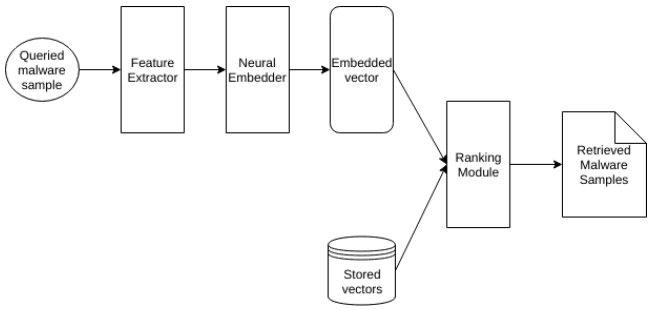

Figure 2: retrieval phase

3.2 Tasks

학습페이즈에서는 IR system 에서 사용하기 위한 벡터 리프레젠테이션을 얻기 위한 Auxiliary Task를 수행한다. Auxiliary Task 는 멀웨어 분류 태스크를 학습하는 것으로 싱글 레이블 클래시피케이션이나 멀티레이블 클래시피케이션이 될 수 있다. f 인 f 는 멀웨어 피쳐로부터 레이블을 추정하는 함수이다. 함수 f 는 다시 g 와 h 의 합성 함수로 정의할 수 있다. h 는 위에서 설명한, 멀웨어 피쳐로부터 vector representation 을 얻을 수 있는 임베딩 함수이다. g는 임베딩으로부터 label 을 추정할 수 있는 클래시파이어이다. f(Z) 리트리벌 페이즈에서는 멀웨어 샘플 쿼리 q를 입력받아 학습 페이즈에서 구했던 h 함수를 이용하여 임베딩한다. 임베딩된 쿼리 eq 와 E 의 원소들 간 거리를 d 함수를 통해 측정하고, 가까운 k 개의 neighbor 를 랭킹 모듈 R 을 통해 반환한다. Result tuples = (Xvecj) where

4 MALWARE IR SYSTEMS

4.1 Desiradata

Submissions must be received at https://ccs17.hotcrp.com/

5 DESIGN AND IMPLEMENT OF MALWARE IR SYSTEM

두 페이즈의 구조도는 Figure 과 같다. 두 페이즈에서 공통으로 존재하는 모듈은 Feature Extractor와 Neural Embedder 가 있다. 벡터 학습 페이즈에는 Classifier 가, 리트리벌 페이즈에는 랭킹 모듈이 포함되어있다.

5.1 Feature Extractor

Feature extractor 는 Malware Rawdata 로부터 Handcrafted feature 를 추출하는 모듈이다. Handcrafted feature로 PE 같은경우에는 Size 와 Entropy, Histogram of API Calls, … 등을[*] 추출한다. APK 같은 경우에는 추가적으로 Permission, … 등의 피쳐를 추출한다.

5.2 Neural Embedder

Neural embedder 는 Feature extractor 모듈에서 추출된 멀웨어의 피쳐들로부터 Representation vector 를 뽑는 모듈이다. theta 로 Parameterized 되어있는 뉴럴 네트워크이며, 파라미터는 벡터 학습 페이즈에서 Auxiliary Task 를 수행하면서 Optimize 된다. 그리고 리트리벌 페이즈에서 파라미터는 freeze 되어 업데이트되지 않는다.

5.3 Classifier

classifier

6 AUXILIARY TASK

6.1 Single label classification

Single label classification 딥러닝 모델로 만든 Malware IR System 장점 : 임베딩된 벡터 representation 에 대해 kNN 해서 IR 이 가능하다. 한계 : 사실 같은 상위카테고리인데 멀리 임베딩된다. 해결법 : 멀티레이블 쓰

6.2 Multi label learning

Multi label classification 딥러닝 모델로 만든 Malware IR System 장점 : 같은 대분류 내의 소분류 애들이 비슷한 곳에 임베딩된다. 한계: inner class variance 가 너무 커서 IR 할 때 distance 계산 시 노이즈가 발생한다. 해결법 : 메트릭러닝인 센터로스를 쓰자

6.3 Mutli label centerloss classification

장점: innerclass variance 가 줄고 interclass variance 가 늘어난다. 한계: 태그 간 관계가 사람의 인지와 다르다 해결법 : 태그 간 중요도를 Constraint 로 넣자

6.4 Mutli label weighted centerloss classification

Mutli label classification + weighted centerloss 딥러닝 모델로 만든 Malware IR System 장점: 태그간 관계가 사람의 인지와 같다. IR 결과가 사람이 생각하는 것과 같이 나온다.

7 RETRIEVAL TASK

리트리벌 페이즈에서는 속도와 효율성이 중요하다. 우리 시스템은 저장된 임베딩 벡터들에 대해 거리를 모두 알아야 하므로 저장 개수가 많을 수록 느려지고, 이를 해결하기 위해서 KNN 대신에 ANN 을 사용하였다.

8 EXPERIMENTS

8.1 Datasets

모델의 학습과 평가에 사용된 dataset은 VirusTotal로부터 crawling 된 샘플 PE 5만여개와 APK 8만여개 중 security expert가 검수하여 확진을 내린 PE 샘플 2천여개 k종의 family, APK 샘플 2만여개 l종의 family로 구성하였다. 또한 VirusTotal에서 제공하는 수십개 이상의 AntiVirus 엔진의 탐지결과로부터 Labeling을 자동화하였다. VirusTotal로부터 각 AntiVirus 엔진들이 탐지해낸 결과를 의미있는 word 단위로 파싱하고, 샘플들로 부터 얻어진 word token들을 대상으로 security expert가 의미있는 토큰과 그렇지 않은 토큰을 선별하는 작업을 수행하였다. 추가로, 의미 있는 토큰에 대해 중요도에 따라 우선순위를 매기도록 하였다. 이를 통해 학습

샘플에 대해 expert가 일일히 수동으로 label을 다는 시간을 줄이면서도 높은 accuracy의 label을 얻을 수 있다. 또한, 하나의 샘플에 single label이 달릴 때에 비해 multi label을 사용 할 경우 악성코드의 다양한 행위 특성을 반영하도록 모델을 학습 시킬 수 있다.

8.2 Neural Network Details

Deep and wide

8.3 Results

표 : Performance measures of Malware IR Systems metric performance measures: acc, auc, tf-idf of labels(?) row : 기존 방법들, single, multi, weighted multi col : single-label classification accuracy, multilabel classification auc non-metric performance measure : top query results querying results sorted by distances

8.4 Visualization

semantic synchronization level Visualization of representations

9 FUTURE WORKS

현재 방법은 멀티 레이블이 다 정확해야 한다. 아니라면 모델이 noise 로부터 오해를 쉽게 한다. 하지만 이 도메인에서 정확한 레이블을 얻기란 쉽지 않다. 따라서 label noise-robust MLC 를 만들고 멀웨어의 feature 로부터 멀웨어 샘플의 리얼 레이블을 추정하고, 이를 통해서 IR 시스템을 만들면 더 좋을 것 같다. 더불어 새로운 샘플에 대한 빠른 대응도 IR 시스템에서 중요한 부분인데, 딥러닝 모델이다보니 학습해야할 시간이 필요하다. continual learning 혹은 online update 를 사용해서 이를 해결하는 것도 우리의 Future work이다.

10 CONCLUSIONS

In conclusion, it is rarely a good idea to include the same section three times in a paper, or to have a conclusion that does not conclude.

A LOCATION

Note that in the new ACM style, the Appendices come before the References.